EINDHOVEN
UNIVERSITY OF
TECHNOLOGY

Eindhoven University of Technology

MASTER

Workflow and process mining in healthcare

Gupta, S.

Award date:
2007

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

TECHNISCHE UNIVERSITEIT EINDHOVEN
Department of Mathematics and Computer Science

Workflow and Process Mining in Healthcare

S. Gupta

Supervisors:
Prof. Dr. Ir. W. M. P. van der Aalst (W&I- AIS)
Dr. A. J. M. M. Weijters (TM-IS)
Dr. A. K. Alves de Medeiros (TM-IS)

Eindhoven, May 2007

Dedicated to

my Parents, OmPrakash and Swati
for being what they are.
&
my husband Shishir
for always being THERE.

Abstract

In this thesis, we explore how process mining techniques can be used to gain insights into the healthcare domain. In this domain one has to cope with cross-functional and multi-disciplinary processes and these are characterized by the terms, dynamic and flexible domain. Today's healthcare organizations are striving to provide timely, cost effective and quality medical services. The first step to achieve this is to analyze the present processes. The process mining research area aims at extracting useful and meaningful information from event logs. In this research project, we work on process mining techniques that automatically discover the process model underlying the existing processes. The performance of process mining techniques on less structured healthcare processes is investigated by using data from two healthcare organizations. A critical evaluation has been made for some of the existing algorithms in the Process Mining Framework (ProM). With a focus on their limitations a new plug-in for the ProM framework for discovering association rules was proposed and also implemented. These association rules in combination with the clustering technique can be further used for generating process models specific to a group of patients sharing some similar characteristic. The focus of this work is to use all these tools and techniques to gain information about the less structured and flexible processes in healthcare.

Keywords: Process Mining, The HeuristicsMiner, the Disjunctive Workflow Schema approach, Weka machine learning library, Association Rules, Apriori, PredictiveApriori, Clustering.

Acknowledgements

This report is the result of my graduation project at the department of Technology Management of Eindhoven University of Technology, section Information Systems (IS). The realization of this project was possible due to the guidance and support of various people. I take this opportunity to deeply thank these people who during this one year provided me with useful and helpful assistance.

First and foremost I would like to thank my supervisor, Wil van der Aalst, who inspires just by what he is. I would like to express my gratitude to him for developing my interest in this area from his teachings for the course: Business Process Management. I also would like to thank my other supervisors Ton Weijters and Ana Karla Alves de Medeiros. Ton has designed the HeuristicsMiner algorithm and he always patiently listened to my problems with it. He is just so easy to talk to and this is what I admire in him the most. Ana Karla Alves de Medeiros was one of those persons who gave feedback on my way of working and took out time to encourage me from her own experiences as a PhD student.

My gratitude also goes out to the various researchers in the J-corridor. Anne Rozinat and Minseok Song for helping me out with the Java problems and, Christian Gunther for helping me start as a ProMdeveloper. Ronny Mans, my then roommate also owes huge thanks for being such a great problem solver and also for many other informal discussions that we enjoyed. The entire J corridor and the process mining group stimulated me to work hard and be something. I found these people amazing, very hard working and very motivating. I really learnt a lot from all these people.

Besides, I would like to thank one of the family friends from India, Anukul Shukla. He spent a lot of time chatting with me on yahoo messenger and discussing my Java problems. He is truly a Java Guru. I would also thank my many Indian friends living in Nijmegen who always patiently listened to my project status and gave me an atmosphere like back home. Thanks a lot Smruti, Ashwini and Angira. My thanks also reach out to Harpreet Bhullar, Stephen van der Elshout, Frank Nabuurs and Tanya Noy for always being such nice friends and understanding me when I said I am busy.

Last but definitely not least, I would like to thank my parents and parents-in-laws in India. My parents who are a source of constant moral support always encouraged me to give my best. Whatever I am good today comes from their unconditional love and faith in me. I could also not forget my brother Saurabh who loves me so dearly and always hoped I finished my project in a nice way. Thanks is also due for my sister-in-law, Pragati for the nice times we spent chatting on the messenger. One of my greatest sources of encouragement was also the mobile messages from my mother-in-law who always penned these messages so beautifully that my heart really said ‘thanks Mummyji’. My father-in-law’s proud in seeing me study also made me happy and kept me going in the toughest times too. My brother-in-law Saurabh also shared my feelings and understood the difficulties I was facing during the project as he himself was going through the same phase in his master studies in India. Finally, I would like to mention about the greatest force behind me, my husband Shishir. He had faith in me even when I lost one in myself, and always kept inspiring me to do my best and finish the project in a great way. He also managed our house nicely when I was busy with my work. I would simply not forget the days when I got everything from breakfast to dinner. Thank you is a small word for him for he has truly been a friend, philosopher and guide.

Shaifali Gupta
Nijmegen, May 2007

1 INTRODUCTION 5

1.1 ASSIGNMENT DESCRIPTION 5
1.1.1 Characteristics: Healthcare domain........ 5
1.2 RESEARCH STATEMENT 6
1.3 RESEARCH APPROACH 7
1.3.1 Case Study1: Catharina Hospital, Eindhoven, the Netherlands 7
1.3.2 Case Study2: Stroke patients in Italian region of Lombardia, Italy........ 7
1.3.3 Research design 8
1.4 THESIS STRUCTURE 8

2 PRELIMINARIES........ 10

2.1 BUSINESS PROCESS MANAGEMENT 10
2.1.1 Definitions........ 10
2.2 WORKFLOW TECHNOLOGY 10
2.3 PROCESS MINING........ 11
2.2.1 Definition/Concept........ 12
2.2.2 Use of Process Mining 13
2.3 PROCESS MINING FRAMEWORK 13
2.3.1 Introduction........ 13
2.3.2 Plug-ins: ProM framework 14
2.4 CONCLUSION 15

3 CRITICAL EVALUATION OF THE HEURISTICSMINER AND THE DWS ALGORITHM........ 16

3.1 CASE STUDY1 16
3.2 THE HEURISTICSMINER ALGORITHM 18
3.2.1 Concept 18
3.3 THE HEURISTICS MINING PLUG-IN 21
3.3.1 The parameters 21
3.3.2 Experimenting with the HeuristicsMiner algorithm........ 26
3.4 EXPERIMENTS WITH THE HEURISTICSMINER 28
3.4.1. Illustration 1........ 28
3.4.2. Illustration 2........ 30
3.4.3. Illustration 3........ 31
3.4.4. Illustration 4........ 33
3.4.4 Summary: Experiments and Observations 35
3.5 THE DISJUNCTIVE WORKFLOW SCHEMA ALGORITHM 36
3.5.1 The DWS plug-in........ 37
3.5.2 Observations 40
3.6 CONCLUSION 40

4 MINING ASSOCIATION RULES OUTSIDE PROM 42

4.1 ASSOCIATION RULES 42
4.1.1 Definitions: Association rules, Support & Confidence 43
4.1.2 Algorithms for Association Rules........ 44
4.2 ASSOCIATION RULES AND WEKA LIBRARY 46
4.2.1 Introduction........ 46
4.3 EXPERIMENTAL RESULTS........ 47
4.3.1. Illustration 1........ 48
4.3.2. Illustration 2........ 49
4.4 CONCLUSION 50

5 MINING ASSOCIATION RULES INSIDE PROM 51

5.1 THE ASSOCIATION RULE MINER (ARM) PLUG-IN........ 51
5.1.1 The features of the ARM........ 51
5.1.2 Interesting rules 52

5.1.3 Parameters: Apriori algorithm *53*
5.1.4 Parameters: PredictiveApriori algorithm *53*
5.2 EXPERIMENTING WITH THE ARM 54
5.2.1 Illustration 1 .... *55*
5.2.2 Illustration 2 .... *56*
5.2.3 Illustration 3 .... *57*
5.2.4 Illustration 4 .... *58*
5.3 PERFORMANCE ISSUES 59
5.4 CONCLUSION 62

6 CLUSTERING 63

6.1 INTRODUCTION 63
6.2 CLUSTERING IN THE ARM 64
6.2.1 Illustration 1 .... *64*
6.2.2 Illustration 2 .... *65*
6.3 CONCLUSION 67

7 EVALUATION 68

7.1 CASE STUDY2 68
7.2 EXPERIMENTAL RESULTS 69
7.2.1 Illustration 1 .... *69*
7.2.2 Illustration 2 .... *71*
7.2.3 Illustration 3 .... *73*
7.3 THE HEURISTICSMINER & THE DWS VS. THE ARM 75
7.4 LIMITATIONS OF THE ARM 77
7.5 CONCLUSION 77

8 CONCLUSION 79

8.1 SUMMARY 79
8.2 INSIGHTS FROM MEDICAL DOMAIN 79
8.3 CONTRIBUTIONS 80
8.4 FUTURE WORK 81

EPILOGUE 82

BIBLIOGRAPHY 83

A: PROCESS MINING FRAMEWORK AND MXML FORMAT 86

B: CASE STUDY1 88

C: A SAMPLE MS-ACCESS TABLE AND ITS CORRESPONDING MXML LOG 100

D: PARAMETERS OF THE HEURISTICSMINER 105

E: MXML LOGS FROM CASE STUDY1 111

F: EFFECT OF THE *ALL-ACTIVITIES-CONNECTED HEURISTIC* HM PARAMETER 117

G: FITNESS OF PROCESS MODELS 119

H: COMPLETENESS AND SOUNDNESS IN THE DWS APPROACH 120

I: PSEUDO CODE FOR ASSOCIATION RULE ALGORITHMS 121

J: THE WEKA LIBRARY 124

K: CASE DATA EXTRACTION MINING PLUG-IN 125

L: USING CSV FROM PROM FOR WEKA 129

M: CASE STUDY2 131

N: MXML LOGS FROM CASE STUDY2 142

O: ASSOCIATION RULES WITH WIDE RANGE OF CONFIDENCE VALUES 152

P: TECHNICAL ARCHITECTURE OF THE ARM 153

1 Introduction

This master thesis is the result of the graduation project for study Master of Science: Business Information Systems at Technical University of Eindhoven (TU/e). The assignment is carried out at the Information Systems group of the Faculty Technology Management, TU/e. The subject of this Master's thesis is to study how the concepts from the areas of Workflow Management (WfM) and Process Mining can be used in healthcare organizations.

1.1 Assignment description

In modern day organizations, information systems and information technologies are becoming all pervasive and there is immense growth in their use. Today's information systems (IS) have no existence of their own unless taken in the context of an organization and its business processes [14]. Such an information system, which is a generic software system, driven by explicit process designs to enact and manage operational business processes is referred to as Business Process Management Systems[1] (BPMS) [1]. These systems record the data on the executed activities in form of event logs. An event log is like a history of what happened in the information system. This recorded data can be helpful to gain a clear picture of underlying business process. The Business Process Management (BPM) systems today are also focusing on Business Process Analysis (BPA), which covers functions of diagnosis and simulation. One of its emerging areas is the Business Activity Monitoring (BAM). The goal of BAM is to collect data at runtime in form of the event logs to support process design and analysis. This information can be used to automatically derive a process model, which can be further analyzed so as to discover its strengths and weaknesses. This approach is known as process mining. BPA includes process mining. This is the focus of the research in this graduation project. The main idea of this thesis is to employ process mining techniques to gain understanding about the processes in the healthcare domain. Before deriving and stating the research goals for this thesis, we first take a look at the characteristics of the healthcare domain in the following sub-section.

1.1.1 Characteristics: Healthcare domain

Patients visit various healthcare organizations to get diagnosis and treatment of their problems. The goal of healthcare organizations is to provide services to their patients in terms of quality, timing and functionality. But this is not as simple as it sounds. In this section we take a look at the characteristics of the healthcare domain.

1. *Dynamic, complex and cross-functional processes*- Healthcare processes involve clinical and administrative processes, large volumes of data and a large number of people, patients and personnel. There are also financial tasks linked with these processes. It is apparent that healthcare processes are just not only related to the health of a patient, but they also involve procedures from other disciplines like management, finance, IT etc. Moreover a treatment process may be dynamic and can become complex. For instance, a patient may visit a hospital for treatment for disease X but during diagnosis or treatment he may develop some other disease like Y, therefore the process of treatment cannot be viewed as a simple sequential process. It may consist of treatment of various illness conditions concurrently and may also involve personnel from various disciplines.
2. *Issues concerning automation, collaboration and coordination*- Most of the activities in each process can only be partially automated as many trade-offs, decisions and actions must be performed by people and cannot be automated or even partially delegated to automated means. Besides automation issues, the quality and degree of collaboration and coordination among humans, and between humans and automated means also play a crucial role in delivering high quality services to the patients [21].
3. *Improper data management*- The management of the hospitals and the patients both suffer from data overload. The data is often redundant, inaccurate, uninformative or confusing. Thus, it is difficult to keep up with the increasing demand for reliable, broadly available health care information.

[1] In Chapter 2 we will elaborate the concepts of BPMS.

4. *Simultaneous functioning of various applications*- There exist many departmental applications that support specific functions, and the data these applications capture often reside within them and is not easily made available to other processes that may require similar data. This also increases data redundancy and also makes the situation confusing because different applications may store same data about the same entity, e.g. a patient's general details can be stored in the applications being used in cardiology, radiology etc. units.
5. *Ad hoc actions and process changes*- In a hospital, actions are influenced by organizational and medical events like introduction of new administrative procedures, medical approaches and technological developments. These events force users like doctors, nurses, and other staff members to change, extend or discontinue the usual procedures. Such unstructured and ad hoc actions are difficult to model and automate. In such cases, it becomes critical and of utmost importance that the changes in the healthcare processes be taken care of [21].

It is apparent that the business processes in the medical domain are dynamic, ad-hoc, unstructured and multi-disciplinary in nature. Also in modern times, healthcare organizations place strong emphasis on medical and organization efficiency and effectiveness to control their healthcare expenditures. They aim at providing world class services at low cost. In such a situation, it becomes of utmost importance to evaluate the existing infrastructure of the services being offered by these organizations. This is where process mining techniques can be employed to extract process models from the event logs [3] of information systems deployed in healthcare organizations. These process models as mentioned earlier can be analyzed to gain an insight into the reality. Process mining techniques help us to understand what is actually going on in reality and if it is what is actually desired.

1.2 Research statement

The research objective of this thesis is to explore how process mining techniques can be used to gain insights into healthcare processes. The characteristics of the healthcare domain as seen in the previous section leads to the following problem definition:

Business processes relating to healthcare procedures like treatment, diagnosis etc. in the medical domain are not easy to understand and automate as they are dynamic, ad-hoc, unstructured and multi-disciplinary in nature.

To be able to understand whether a healthcare organization achieves its goals of providing timely, cost effective and quality medical services we need to analyze its healthcare processes. In this context we used empirical data (i.e. the recorded histories of process events that occurred over time in a healthcare process) to experiment with some of the process mining algorithms available in the Process Mining (ProM) framework[2]. The focus was on the ability of these algorithms to capture the underlying process. The first research goal is defined as follows:

Study and evaluate the performance of the process mining algorithms on the healthcare processes.

As the focus of these algorithms is to study the control-flow i.e. the execution order of tasks in a healthcare process, we chose one of the robust algorithms implemented in ProM: the HeuristicsMiner algorithm (HM). This algorithm generates a process model underlying an event log based on causal dependencies[3] between tasks and is robust to noise and imbalance. This leads to our second research goal:

Analyze the process models derived from the HeuristicsMiner algorithm and evaluate them on criteria such as simplicity, ease of understanding and the degree of meaningful information obtained from them.

The HM is available as the Heuristics Mining plug-in in the ProM framework. A disadvantage of this algorithm is that it generates complex spaghetti-like models (huge and confusing) for less structured

[2] ProM framework available at http://prom.processmining.org, is a generic tool providing various process mining algorithms.

[3] Causal dependencies will be explained in Chapter 3, Section 3.2.1.1.

processes. These models also show some inconsistencies in form of missing dependencies and tasks. Therefore, it was decided to explore another process model representation in form of rules rather than a visual process model. In this context, the Disjunctive Workflow Schema (DWS) plug-in of ProM was evaluated. This plug-in provides different variants of an overall workflow schema based on some rules. For generating process models for different clusters it uses the HM algorithm. But the plug-in has disadvantages of its own as well the problems of the HM algorithm. The complex results obtained from these plug-ins led us to explore a technique that derives simpler models for complex domain of the healthcare. In this context, *association rules*, an established technique in classical data mining field were found interesting for further investigation. These rules are simple and easy to comprehend. In this thesis we would investigate the use of association rules in overcoming the limitations of the HM algorithm and the DWS approach. For this, the Case Data Extraction (CDE) plug-in available in ProM was used in combination with the Weka machine learning library (Weka)[4]. The output of the CDE was given to the association rule algorithms implemented in Weka. The association rules would help us in obtaining simpler and understandable process models that give meaningful insights into the underlying process. Therefore, the third research goal is:

Investigate the usefulness of Association Rules as an alternate process model representation.

Besides investigating and utilizing the concept of association rules as an alternate process model representation, it would also be seen how they can help us group entities such as patients, treatments, complications etc. in the healthcare domain. For example, these rules can be used to generate homogeneous groups of patients, treatments, complications etc. to obtain process models specific to these groups. This leads to the fourth research goal:

Develop a mechanism to use Association Rules for clustering different patient (or complications, treatments etc.) groups into one homogeneous group.

1.3 Research approach

To understand how process mining techniques can be used in gaining a better understanding of processes within a healthcare organization, we stated our research goals in the previous section. These research goals serve as a directive guide to complete this research assignment. In the thesis we experiment with data from two healthcare organizations. In this section, we introduce these organizations and describe the approach that we followed to achieve our research objective and goals.

1.3.1 Case Study1: Catharina Hospital, Eindhoven, the Netherlands

The data obtained from the Catharina Hospital, Eindhoven describes the various activities that take place in the Intensive Care Unit (ICU) of the Catharina hospital. It contains records mainly related to patients, their complications, diagnosis, investigations, measurements, and characteristics of patient's clinical admission (his specialist, date of admission, main and secondary diagnosis, treatment, etc.). The data pertains to 23779 patients. This data may be used for obtaining information about the control-flow, resources (doctors, specialists etc) performing various tasks, the interrelations between various resources etc. This data is used for experiments with the HM algorithm and other algorithms implemented in the ProM. Chapter 3 elaborates on the characteristics of this data.

1.3.2 Case Study2: Stroke patients in Italian region of Lombardia, Italy

A preliminary study was conducted in 4 districts in the northern Italian region of Lombardia on patients of acute stroke[5] and transient ischemic attack on first-ever stroke patients. It aimed at studying the effect of the American Heart Association guidelines on 386 such patients. We would like to thank Dr. Silvana Quaglini (Università degli Studi di Pavia, Italy), Dr. Anna Cavallini (IRCCS C. Mondino, Pavia, Italy), and Dr.

[4] Weka is an open source tool for data mining tasks developed at the University of Waikato, New Zealand and is available at http://www.cs.waikato.ac.nz/ml/weka.

[5] Acute stroke is a vascular condition that precipitates neurological damage and is the second leading cause of death in industrialized countries [17].

Giusseppe Micieli (IRCCS Humanitas, Rozzano, Italy) for providing us with the database of this study to test the results. The database contains information about the acute phase to the sub-acute phases of the patients suffering from stroke. These records contain data about: (1) the baseline characteristics: age, sex, living conditions, drug use before stroke etc.; (2) vascular risk factors and conditions like hypertension, diabetes mellitus, smoking, alcohol consumption etc.; (3) clinical condition at admission: level of consciousness, level of disability etc.; and (4) diagnostic tests, amount of inpatient rehabilitation etc. [18]. Chapter 7 elaborates on the characteristics of this data. This data is used for experiments (cf. Chapter 7) with a newly implemented plug-in for mining association rules from an event log. Hence this dataset is also used for cross validation of our approach.

1.3.3 Research design

To achieve the research goals, we realized that the starting point is to gain understanding about the real world data from the healthcare domain. Therefore we started with using the data from Case study1 to learn the characteristics of the healthcare data and experiment with the HM available in ProM. In this thesis we use process mining techniques to induce models from data from hospital information systems and to gain as many insights as possible about the investigated process, therefore we can state that the approach is an inductive research approach. This part of the thesis focuses on the second research goal. As will be seen in practical situations, process mining algorithms generate spaghetti-like models, which are very complex and difficult to comprehend. So we search for some alternatives. The alternative that we suggest is the use of data mining techniques[6] to induce rules that can be used to construct process models. Here the focus is on the third research goal. Further, these rules can also be used in clustering the event logs. The goal of clustering is to obtain homogeneous group of patients. This focuses on the fourth research goal. Once these clusters are obtained, various process mining algorithms can be used for construction of simpler and understandable process models. We evaluate our research and development work using Case study2. This is intended to show that our proposed method can be generalized and that it results in useful insights into the considered process.

This research design led to the thesis structure defined in the next section. Figure 1.1 shows the research design along with the thesis structure.

1.4 Thesis structure

In this chapter, we have given the assignment background and the motivation for this research. We also defined the research goals and the approach that will be followed to achieve them. The remainder of this thesis is structured as follows: Chapter 2 presents the preliminary and detailed knowledge about workflow, workflow management and process mining along with the description of the ProM tool. This is necessary to understand the work done in this thesis. Chapter 3 focuses on the first and second research goals and uses the data from the Case study1. Here, the HM and the DWS algorithms are critically evaluated. Chapter 4 and 5 focus on achieving the third research goal and introduce alternative process model representation in form of association rules. In Chapter 5, we propose the implementation of a new plug-in in the ProM tool to generate association rules from the event logs. Chapter 6 focuses on the fourth and last research goal. Here we describe how the concept of clustering is used and implemented in the new plug-in described in Chapter 5. In Chapter 7, this new plug-in is evaluated using Case study2 and the observations are summarized. Chapter 8 concludes the research work with a discussion about its contributions. We also give suggestions for future work.

[6] Data mining concepts are explained in Chapter 2.

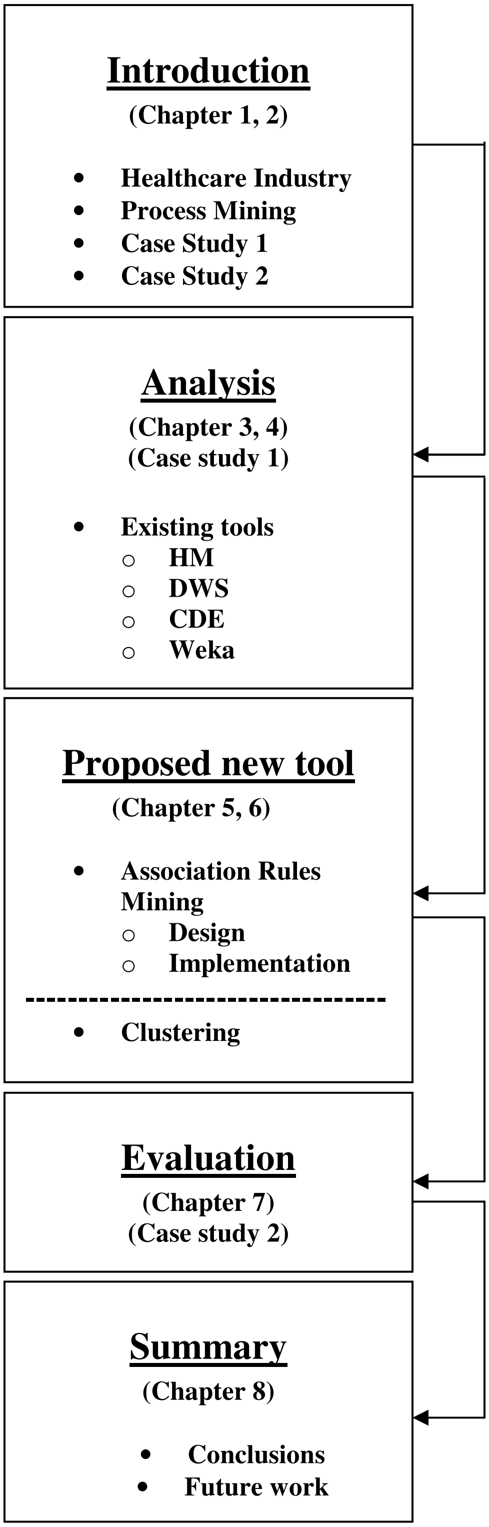


Figure 1.1: Research design

2 Preliminaries

This chapter provides some background information on business processes, workflow and process mining that are important and useful to understand the remainder of this thesis.

2.1 Business Process Management

We start this section by first understanding the term business process. After this we acquaint ourselves with the concepts of BPM and BPMS.

2.1.1 Definitions

Business process refers to how an organization has decided upon its flow of activities so as to produce desired results by making optimum use of its resources in form of raw material, personnel and their skills, and equipments. Business processes can be formally defined as [21]:

> *"Sets of partially ordered and coordinated activities, often cutting across functional boundaries, by which organizations accomplish their missions."*

The purpose of any business process is production of products. These products may be tangible like a car, as well as intangible like a service. Services include handling of insurance claim, treatment or even assessment of a scientific paper. The concept of managing business processes is referred to as BPM. Van der Aalst defines it as [6]:

> *"Supporting business processes using methods, techniques, and software to design, enact, control and analyze operational business processes involving humans, organizations, applications, documents and other sources of information".*

The software systems that manage business processes are known as BPMSs and defined as [1]:

> *"An information system, which is a generic software system, driven by explicit process designs to enact and manage operational business processes is referred to as Business Process Management Systems."*

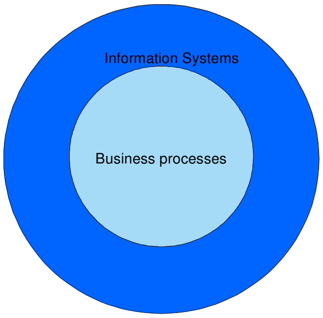


Figure 2.1: BPMS-Information systems with business processes at the core

Processes in organizations can only be modelled and automated if they are repeatedly performed in the same way i.e. they have a distinct structure. The automation of business processes is thus based on explicit modelling of processes and organizations [29]. This is exactly the concern of workflow technology.

2.2 Workflow technology

Workflow refers to the tasks, resources and triggers associated with a specific process. Workflow management systems (WFMS) are the systems implementing workflow technology. They are a type of BPMS. Some of the examples of the WFMS include Staffware, Cosa and MQSeries. The leading enterprise resource planning systems like SAP, Baan, PeopleSoft, Oracle and JD Edwards also offer a workflow management module [1].

WFMS provide an environment to automate and assist in the management of tasks and the flow of work-items from one task to another [37]. These systems require a process model and their main function is to ensure that all the activities are performed in the right order and by the right resource. Let us understand workflow technology in context of healthcare.

Workflow systems at runtime invoke instances of the business process automated and configured through it. In the healthcare domain, this instance is a 'case'. This case is an object, which changes states as the process goes from one stage to another. In context of medical domain each patient is an individual case and a business process may be a process of treatment, medical examination etc. These processes are composed of different tasks. They may be fully or partially automated or may require a combination of both-humans and computers. These tasks are placed in a queue so that they can be undertaken by the humans. This queue is known as a work-list and each item in such a list is a work-item.

Formally a WFMS provides means to [29]:

- Model the processes in terms of activities and state-transitions,
- Model the organization in terms of organizational units and workflow participants,
- Match workflow participants and activities and,
- Bring the processes into action and provide worklists for the application systems.

The focus of traditional workflow management systems is typically on the design and configuration phase of the BPM life cycle given in Figure 2.2 below:

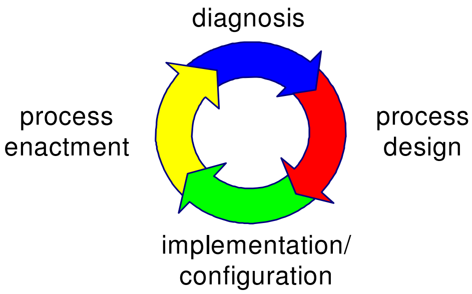


Figure 2.2: Phases: The Business Process Management Life cycle

- **Process design phase**- In this phase operational processes are designed or redesigned.
- **System configuration phase**- The design made in the previous phase is implemented by configuring an information system like Workflow Management System (WFMS).
- **Process enactment phase**- This is the phase of execution. The process, which is configured, is actually executed using the information system.
- **Diagnosis**- The diagnosis phase is like a feedback phase where the process and the system configuring it is analyzed for problem identification and seeking improvements.

Though WFM systems record the data about the activities executed in form of event logs, they do not provide much support for their analysis. BPM systems are focusing more on analysis. In the next section, we discuss the concept of process mining as the one of the techniques for analyzing business processes.

2.3 Process Mining

It is already indicated that WFM systems do not provide much support for analysis of event logs and systems ignored issues of monitoring, simulation, flexibility, diagnosis etc. In this section process mining is discussed as a technology to contribute to these neglected issues.

2.2.1 Definition/Concept

Due to little focus on diagnosis, many workflow projects failed and hence today workflow vendors are positioning their systems like BPM systems [5]. Business Activity Monitoring (BAM)[7] is one of the emerging and widely popular areas in Business Process Analysis (BPA). BAM is an automated form of process monitoring. BAM tools use data logged by the information systems to monitor and analyze processes in an organization. This approach is known as process mining. Process mining as the name suggests is a technique for mining a process. It can be defined as [5]:

"Process mining is the method for distilling a structured process description from a set of real executions."

In the above definition, 'a set of real executions' means a process log with data about the order in which the activities were executed. It is like a history of what happened in the process. This set of executions is also referred to as an 'event log', 'history' or 'audit trail'. An event log contains information about events which refer to 'activities' or 'tasks' executed in a particular process and for a specific 'case'. The case is the object being handled. Typically event logs also record the time when these tasks were executed or when they were in a particular state. This is known as the timestamp of an event. Event logs also store information about the originator of a task, i.e. who performed which task or initiated an event. Table 2.1 given below shows an example event log.

Table 2.1: An example event log

Case ID	Activity	Originator	Timestamp
Case 1	Register	Samantha	18-10-2006:12:17
Case 2	Register	Peter	18-10-2006:12:26
Case 1	Evaluate	Jo	18-10-2006:12:36
Case 3	Register	Samantha	18-10-2006:15:10
Case 1	Reimbursement	Andrew	19-10-2006:09:50
Case 3	Evaluate	Jo	19-10-2006:12:56
Case 3	Reimbursement	Andrew	19-10-2006:16:04
Case 2	Evaluate	Jo	20-10-2006:11:34
Case 2	Reimbursement	Andrew	20-10-2006:19:50
Case 1	Reimbursement	Andrew	22-10-2006:08:50

Event log in Table 2.1 depicts history of an insurance information system. The object being handled is an insurance claim and it goes through stages (tasks) of registration, evaluation and reimbursement or cancellation. Column 3 shows the person who performed the corresponding task indicated by the column 2. We can also see the time when these tasks were performed in the system.

An event log consists of one or more Audit Trail Entries. An Audit Trail Entry (ATE) indicates the case id, activity id, originator, timestamp and some other data attributes (if any). It is clear that a single ATE refers to a single case. A case is also known as a process instance. One ATE can not indicate two cases; however there can be several audit trail entries for a case. Following figure shows a single ATE:

Case 3	Evaluate	Jo	19-10-2006:12:56

Figure 2.3: An example ATE representing that 'Jo' performed the task 'Evaluate' for 'Case 3' on '19:10:2006:12:56'

Once we have access to such set of executions i.e. event logs they can be used for gaining insight into a process execution. This means we can take a look at what happened in between the 'start' and the 'end' of a

[7] BAM is the aggregation, analysis, and presentation of real time information about activities inside organizations and, involving customers and partners. (http://en.wikipedia.org/wiki/Gartner)

process. This will help us extract valuable information about how actually the process was carried out. This can be considered as a tracing back to check what happened. This is important because, although the process is carried out as per the specification of the process model, in case of problems emerging during execution workers tend to bypass the system and work “behind its back”. This makes it hard to spot the problems located in the process model itself, as they are effectively evaded by experienced workers, and also other problems arising from these actions can no longer be traced back to the process model and leads to mysterious and unforeseen behaviour.

Following this discussion, it can be followed that process mining is a three-phase process: pre-processing, processing and post-processing [17]. In the pre-processing phase, the event log is read into the ProM and the order between tasks is inferred. In the processing phase, a mining algorithm is applied to this event log and the ordering relationships between tasks serves as the input. For the post-processing phase, both the event log and the generated process model serves as input. They can be used to find additional information about the process i.e. we can now fine-tune the process model as well as show it graphically during the post-processing phase.

2.2.2 Use of Process Mining

Process mining begins with information about an executed process collected by information systems, in the form of event logs rather than with a process design. It is not just about discovery of process models to trace back inefficient behaviour or to find deficiencies in a process but can result into much varied outputs. Below we mention some areas where process mining tools and technique can be used [11]:

- *Process discovery:* Process mining helps to discover the process model by inferring the ordering relations between various tasks in the event log.
- *Delta analysis:* Process mining tools also helps answering the question: “are we doing what was specified?”
- *Performance analysis:* Performance analysis includes the measures that can be used for improvement of process model and their properties.
- *Social network and organizational mining:* Process mining does not only extract the process model, and other parameters like flow times, sojourn times etc. but also includes discovering relationships between the various events and their originators. We can also discover an organizational structure in terms of an activity role-performer diagram, or a socio-gram based on the transfer of work.

In the next section, we introduce Process Mining framework as a platform of process mining algorithms and tools developed by researchers at IS group of TU/e.

2.3 Process mining framework

2.3.1 Introduction

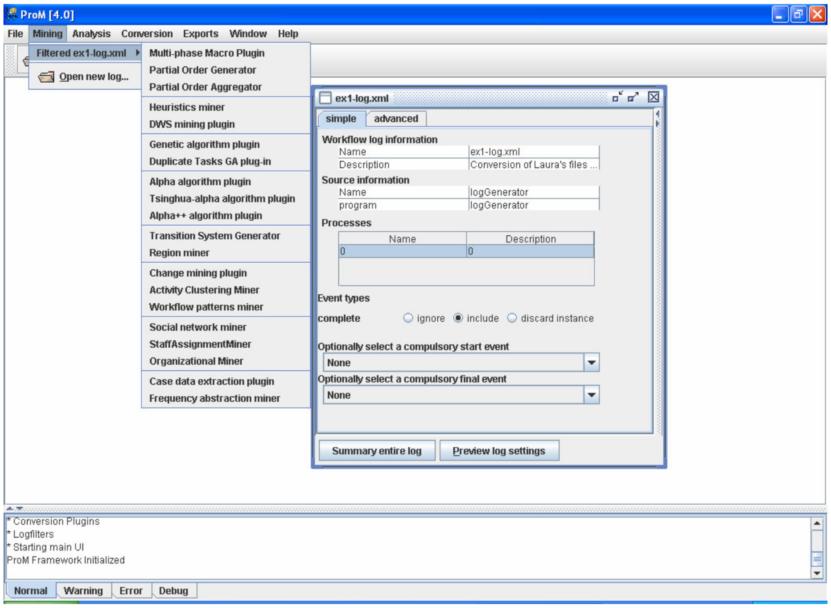
ProM is an extensible framework that supports a wide variety of process mining techniques in the form of plug-ins. The plug-ins are the implementation of various process mining algorithms. ProM is platform independent as it is implemented in Java and is open source.

An event log from an information system is the input to the ProM framework. As mentioned in Section 2.2.1, process mining is a three phased process. In its processing phase a specific plug-in representing a distinct functionality is used depending on the desired output. This output can be in different forms and also can be represented graphically.

The ProM framework accepts event logs from various information systems as its input. It could come from a workflow system like Staffware, Oracle BPEL etc. or from simulation tools such as ARIS, EPC tools etc., so it was required to standardize this input to be supplied to the framework. This common format for the input is known as Mining Extensible Markup Language (MXML) format. As the name indicates, the format is XML based and is defined by an XML schema. The format is described by the document type definition

(DTD) that can be found at [10]. Several adaptors to map logs from different information systems into MXML format were then built in the context of the ProMImport framework[8]. Readers are referred to the Appendix A for detailed reading about the MXML format. MXML is a tool-independent format to log events and can be generated from audit trails, transaction logs and other data sets describing business events.

When the ProM framework was being developed, it focused and provided algorithms only for the discovery of process models. But now it offers functionality for delta analysis, performance analysis and organizational/social network mining too. For example, it can be used for the analysis of event logs, for conversion of a model into another, exporting a model to a file, to find social network between different originators of a process, to understand how properties of a case affect the control flow etc. Figure 2.4 shows a screen shot of the ProM. The event log file which is used for this example can be identified in the figure as ex1-log.xml. In the menu bar mining, analysis, conversion, and export functionalities are offered. The figure also shows, for example the mining plug-ins currently available in the framework. In the next sub-section we take a look at the functionalities offered by the ProM framework. This will help us to form a knowledge base required to understand the evaluation and implementation work done in this thesis.



Workflow log information

Name	ex1-log.xml
Description	Conversion of Laura's files ...

Source information

Name	logGenerator
program	logGenerator

Processes

Name	Description
0	0

Event types

complete ○ ignore ● include ○ discard instance

Optionally select a compulsory start event: None

Optionally select a compulsory final event: None

```
* Conversion Plugins
* Logfilters
* Starting main UI
ProM Framework Initialized
```

Figure 2.4: *Pro*cess *M*ining (ProM) framework

2.3.2 Plug-ins: ProM framework

The plug-in concept of ProM allows for the addition of new functionality simply by adding a plug-in rather than modifying the source code. Version 4.0 of ProM supports 157 plug-ins. Figure 2.5 shows an overview of the ProM framework. As we can see in this figure, the architecture of ProM allows for five different types of plug-ins [26]:

1) *Mining plug-ins* take an event log and produce a process model by implementing some mining algorithm. Some of the examples are the Alpha algorithm, HM, Genetic algorithm plug-in, Region miner, DWS plug-in etc. The language in which these plug-in represent the discovered process model is different. Alpha algorithm expresses the mined process model in terms of Petri net whereas the HM uses heuristics net for the representation.

[8] http://promimport.processmining.org

2) *Analysis plug-ins* analyses a process model. They perform some kind of analysis like Petri net analysis, checking a Linear Temporal Logic (LTL) property, decision point analysis, fitness analysis etc. Petri net analysis includes calculation of invariants, construction of reachability graphs etc. We can check for temporal properties or properties like four-eye principal using the LTL checker plug-in. Many other kinds of analysis plug-ins are available. The input to these analysis plug-ins can be event logs or the process models. This means they can directly be applied to the event logs or they can be used on the process models generated by some mining plug-in or imported from some information systems using import plug-ins.

3) *Export plug-ins* offer "save as" functionality for some objects like graphs. It helps in exporting a model to a file in form of event-driven process chains (EPCs), Petri nets, spreadsheets, grouped XML log files, heuristics net, yet another workflow language (YAWL) files, protos, coloured Petri nets (CPN) etc. These files then can further be used for different kinds of analysis.

4) *Import plug-ins* import a process model from a file and possibly use a log to identify the relevant objects in the model. Import plug-ins makes it possible for the ProM framework to work with a variety of existing systems like EPC Tools, ARIS, Protos, NetMiner etc.

5) *Conversion plug-ins*, as the name indicates, convert one type of model into another. For example it is possible to convert a heuristics net to Petri net and EPC, YAWL models to EPC etc.

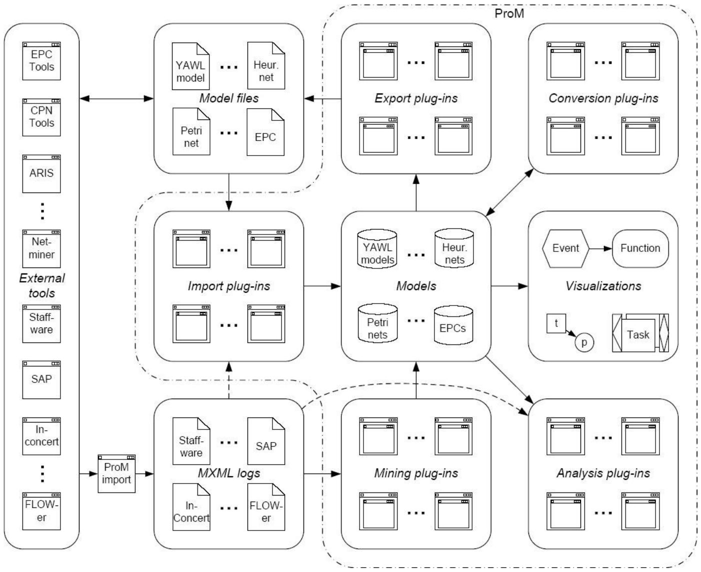


Figure 2.5: Overview of the ProM Framework

Section 2.3 covered general introduction to the process mining tool: the ProM framework. We also saw different kind of functionalities it offers to a user in form of various plug-ins. The ProM framework and literature about its plug-ins can be downloaded from the process mining website: www.processmining.org.

2.4 Conclusion

In this chapter we presented preliminary concepts needed to understand this thesis. We started by introducing business processes, BPM and BPMS. Workflow technology and process mining concepts were also introduced. In the coming chapters these are extensively used and therefore this chapter forms an important starting point for the remainder of this thesis. The next chapter focuses on the second research goal. It includes a detailed description of Case study1 and experiments with the HM and the DWS algorithms in order to get insight into the process model underlying the healthcare processes in Case study1. The chapter also motivates the choice for these algorithms.

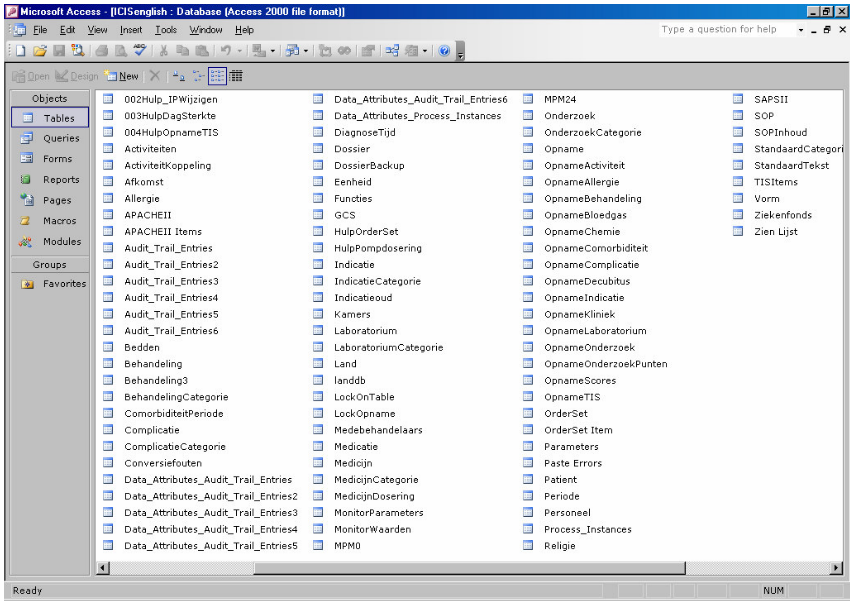
3 Critical evaluation of the HeuristicsMiner and the DWS algorithm

The healthcare domain as discussed in Section 1.1.1 is dynamic, complex and involves various disciplines. In Chapter 2 it was mentioned that it is difficult to automate processes from such domains because of their interdisciplinary and dynamic nature. Moreover, it is also critical to keep a check on the automated processes to produce the expected results in form of quality-and timely healthcare services. Delivering these services is a complex business. Patients are generally older, and tend to have complex medical problems [39]. The drive in healthcare organizations is to reduce costs and at the same time improve the quality of services the patients need. The benefit of workflow management systems in such domain includes cost reduction, improved operational efficiencies, clinical error reduction, improved patient care, better communication and collaboration and real time audit of processes [39]. As we are aware, such systems need a process model, and this process model should clearly depict the control-flow of the tasks in any business process. Therefore the focus of this chapter is on the process discovery. For this, two plug-ins implemented in ProM have been used: the HM and the DWS plug-in.

In real life processes like the Case study1 a lot of noise is expected because of human errors, incompleteness of data etc. and therefore the HM was chosen to investigate the processes in Case study1 because it is one of the robust algorithms till date. The choice for the DWS algorithm was made because it provides process models specific to a group of similar patients present in the log. This would help us to explore smaller and specific healthcare process models. The focus of experimentation with these algorithms is to obtain a process model for the investigated healthcare process as well as to evaluate the performance of these plug-ins for the healthcare data. The structure of this chapter is as follows. Section 3.1 introduces Case study1. Section 3.2 elaborates the fundamentals of the HM algorithm and Section 3.3 explains its implementation in the form of a mining plug-in. Section 3.4 details our experiments with it focussing on the second research goal. The DWS approach is explained in Section 3.5. Section 3.6 concludes the chapter by summarizing the findings.

3.1 Case study1

Case study1 refers to the data obtained from the Catharina hospital. This data is received in form of the database in Microsoft Access. It contains various tables recording information about patients, complications, treatment procedures, various medical tests, hospital personnel, hospital infrastructure like rooms and laboratories etc. The following figure shows a list of tables contained in this database:



002Hulp_IPWijzigen
003HulpDagSterkte
004HulpOpnameTIS
Activiteiten
ActiviteitKoppeling
Afkomst
Allergie
APACHEII
APACHEII Items
Audit_Trail_Entries
Audit_Trail_Entries2
Audit_Trail_Entries3
Audit_Trail_Entries4
Audit_Trail_Entries5
Audit_Trail_Entries6
Bedden
Behandeling
Behandeling3
BehandelingCategorie
ComorbiditeitPeriode
Complicatie
ComplicatieCategorie
Conversiefouten
Data_Attributes_Audit_Trail_Entries
Data_Attributes_Audit_Trail_Entries2
Data_Attributes_Audit_Trail_Entries3
Data_Attributes_Audit_Trail_Entries4
Data_Attributes_Audit_Trail_Entries5
Data_Attributes_Audit_Trail_Entries6
Data_Attributes_Process_Instances
DiagnoseTijd
Dossier
DossierBackup
Eenheid
Functies
GCS
HulpOrderSet
HulpPompdosering
Indicatie
IndicatieCategorie
Indicatieoud
Kamers
Laboratorium
LaboratoriumCategorie
Land
landdb
LockOnTable
LockOpname
Medebehandelaars
Medicatie
Medicijn
MedicijnCategorie
MedicijnDosering
MonitorParameters
MonitorWaarden
MPM0
MPM24
Onderzoek
OnderzoekCategorie
Opname
OpnameActiviteit
OpnameAllergie
OpnameBehandeling
OpnameBloedgas
OpnameChemie
OpnameComorbiditeit
OpnameComplicatie
OpnameDecubitus
OpnameIndicatie
OpnameKliniek
OpnameLaboratorium
OpnameOnderzoek
OpnameOnderzoekPunten
OpnameScores
OpnameTIS
OrderSet
OrderSet Item
Parameters
Paste Errors
Patient
Periode
Personeel
Process_Instances
Religie
SAPSII
SOP
SOPInhoud
StandaardCategori
StandaardTekst
TISItems
Vorm
Ziekenfonds
Zien Lijst

Figure 3.1: Tables in the hospital database

Readers are referred to Appendix B for taking a look at each of the table in this database along with the data contained in them. Contents of some of these tables are discussed below:

- Patient: The table contains personal characteristics of each patient visiting the hospital for some treatment or consultation. Patient's name, date of birth, contact details, gender, insurance details, nationality, weight, religion etc. is recorded in this table. This table contains generic information, common for all the patients visiting the hospital.
- Opname: This table also contains information about a patient, but this information is about the patient's admission to the hospital. This includes the patient's date and time of admission, his admission id (each patient is given a unique identifier every time he visits the hospitals), his room number and ward in the hospital, his doctors, his discharge details etc. It also contains generic information, common to all patients admitted to the hospital for treatment or diagnosis.
- Indicatie: The table Indicatie stores details about indication of a patient's diagnosis. It stores the indication category, diagnosis type, and other related information.
- Personnel: Details like name, contact details, date of birth, department, username and password of the staff members of the hospital is contained in the table Personnel.
- Complicatie: Information about the complications, their category, specialist etc. is stored in this table.
- OpnameBehandeling: This is the table maintaining records of treatment(s) given to a patient.

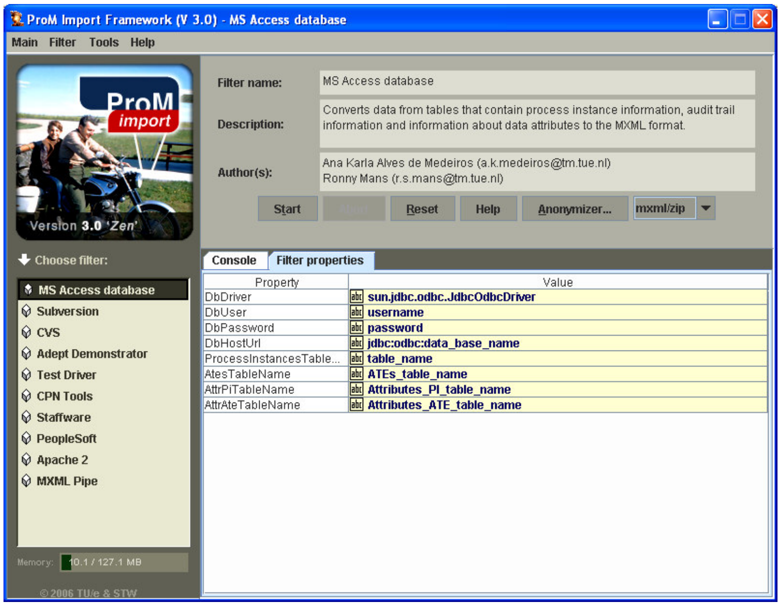
Besides taking an overview of these tables, we also made note of some observations about the data contained in the database. These observations are summarized below:

- In the database, data is categorized under numerous headings (fields) but not all of these contain data. For example, there are numerous complications that any patient can suffer from but the number of patients suffering from these many complications is very low. In process mining terms it means that the number of events per case is very less than the total number of events in the log. This shows that a lot of highly low-frequent events are present in the event log.
- Due to a high degree of low-frequent behaviour it is uncertain if this is due to some human error (erroneous insertions or non-insertions of events) or it is actually low frequent. The possibilities of noise in real world databases like Case study1 cannot be ignored but this possible presence of noise cannot be distinguished from actual process characteristics.

After having an overview of the data and its organization in Case study1, it is understood that these tables cannot be directly used for experimentation in ProM as it accepts data only in MXML format. The conversion of MS-Access data to MXML is achieved by the MS-Access import plug-in (Figure 3.2) implemented in the ProM Import framework[9]. Detailed explanation about this import plug-in is not the focus of this thesis, so the readers are referred to [38] to read more about this conversion process. In Appendix C, an example MS-Access table from Case study1 and its corresponding MXML log can be found.

The logs used for various experiments pertain to the different events that occur for numerous patients. For example, some logs represent the route followed by complications for different patients, and some logs may pertain to various treatment activities etc. In the coming sections (Section 3.2 and 3.3) the algorithm behind the Heuristics Mining plug-in is explained and is evaluated on the basis of its performance on the healthcare data.

[9] We would like to thank the authors (cf. Figure 3.2) of this plug-in for their work.



Filter name: MS Access database

Description: Converts data from tables that contain process instance information, audit trail information and information about data attributes to the MXML format.

Author(s): Ana Karla Alves de Medeiros (a.k.medeiros@tm.tue.nl)
Ronny Mans (r.s.mans@tm.tue.nl)

Property	Value
DbDriver	sun.jdbc.odbc.JdbcOdbcDriver
DbUser	username
DbPassword	password
DbHostUrl	jdbc:odbc:data_base_name
ProcessInstancesTable...	table_name
AtesTableName	ATEs_table_name
AttrPiTableName	Attributes_PI_table_name
AttrAteTableName	Attributes_ATE_table_name

Figure 3.2: MS-Access tables can be converted to the MXML format

3.2 The HeuristicsMiner algorithm

The HM algorithm focuses on the control flow perspective and generates a process model in form of a Heuristics Net for the underlying event log. The formal approaches like the α-algorithm[10] (an algorithm for mining event logs and producing a process model) presupposes that the mined log must be complete and there should not be any noise in the log. However, this is not practically possible. Also, this algorithm does not make use of any frequency information (frequency of various dependencies of the tasks in an event log), which can be quite useful in situations of noise. Readers can refer [11] for detailed reading about the α-algorithm including its limitations. Therefore, the HM algorithm was designed to make use of a frequency based metric and so it is less sensitive to noise and the incompleteness of logs. In the next section the basic concept of this algorithm are explained

3.2.1 Concept

The HM algorithm is a three step algorithm [11, 28]:

1. Construct a dependency graph on the basis of the event log.
2. For each task in the event log establish the input-output expressions in form of type of dependencies between activities, and
3. Discover the long distance dependency relations.

3.2.1.1 Constructing a dependency graph

For mining control flow perspective based on an event log, process mining algorithms analyze the log for dependencies between the tasks. But only the discovery of dependencies is not sufficient, we must also be certain of them. To depict these dependencies the notation taken from [28] is used:

Let W be an event log over a set of activities, T. Let *a*, *b* be activities belonging to the set T. Then we define following dependencies between them:

1. Activity *a* is directly followed by the activity *b* at least once in the log. This relationship is expressed as $a >_w b$.
2. Activities *a* and *b* are in a direct dependency relation i.e. $a \rightarrow_w b$ when *a* is directly followed by *b* but *b* is never followed by *a* i.e. $a >_w b$ happens and not $b >_w a$.

[10] For detailed reading about the Alpha algorithm, please refer [10].

3. Activities *a* and *b* exhibit parallel behaviour if both *a*, *b* follow each other directly and in any order i.e. both $a >_w b$ and $b >_w a$ happens. The notation for this is $a \|_w b$.
4. The absence of any relation between activities *a* and *b* is depicted as $a \#_w b$ i.e. neither $a >_w b$ nor $b >_w a$ happens.

The first step in the HM is the construction of a dependency graph depicting these dependencies. It also depicts how certain we are of a dependency relation. For this the algorithm uses a frequency based metric known as the dependency measure. The formula for dependency measure between two tasks *a*, *b* is given below:

$$a \Rightarrow_w b = \left(\frac{|a >_w b| - |b >_w a|}{|a >_w b| + |b >_w a| + k} \right)$$

Equation 3.1: Dependency measure between a and b

This formula represents that *a*, *b* are two activities in an event log W, $|a >_W b|$ is the number of times activity a follows activity *b* in W and $|b >_W a|$ is the number of times activity *b* follows activity a in W. 'k' is the parameter called dependency divisor (cf. Section 3.3.1.3) and 'k' є N and 'k' >0. The value calculated by this formula is the dependency measure between activities a, b. To understand the significance of the dependency measure, let us assume that in an event log activity *a* is directly followed by *b* in 7 traces but vice versa does not happen. On calculating the value of $a =>_W b$, we have (7-0)/(7+0+1)=0.875. But let us assume that $a >_w b$ occurs in 10 traces, then the value of $a =>_W b$ is 0.90, this indicates that we are more certain of the dependency between *a*, *b*. The default value of dependency measure is between -1 and 1. Once the values for dependency measure are obtained for different pairs of activities in a log, the correct dependency relations between them can be searched. Below we take a look at how to construct a dependency graph.

As we already mentioned, dependency graph is based on the dependency values for different activities, so let us assume that we have a matrix for these values between different pair of activities.

Table 3.1: Dependency value matrix

$=>_w$	**A**	**B**	**C**	**D**
A	0.0	0.975	0.983	0.0
B	0.0	0.0	0.0	0.975
C	0.0	0.0	0.0	0.983
D	0.0	-0.975	-0.983	0.0

Now to generate the dependency graph from above matrix, the *all-events-connected heuristic* is applied. The '*all-events-connected-heuristic'* means that each non-initial activity must have at least one other activity that is its cause and each non-final activity must have at least one dependent activity, and based on this heuristic this matrix is analyzed. We perform following steps:

1. Look for a column that does not have any positive value, clearly it is A. It is our initial activity.
2. Search row A to find what activities depend on A. We see that the values for B and C are high. We choose C as it has the highest value in the row.
3. To find the cause of C, we look in column C and find A as its cause.
4. Now to find what activities depend on C, we search row C. We find D is the depending activity of Q.
5. Now we take activity B. We first find its cause by looking at column B. We can clearly see it is A.
6. To find what other activities depend on B we search its row. We can see that D is the depending activity of B.
7. Combining all this information we draw a graph with these activities A, B, C and D as nodes (cf. Figure 3.3). The activities are shown in a box along with their frequencies in the event log. Arcs connect these activities to one another and have two numbers on them. The first one indicates how reliable is the dependency relation between the corresponding activities and the second one

indicates the number of times this connection exists in the event log. The frequency of activities is depicted in their respective box, e.g., task A executes 100 times, B executes 40 times etc. We also see for e.g. the connection A to B is followed 40 times, and similarly the connection C to D is executed 60 times etc. The reliability of a dependency relation i.e. dependency measure is the first number on the arc, for e.g. the reliability of the task A directly followed by task B is 0.975, the reliability of connection A directly followed by C is 0.983.

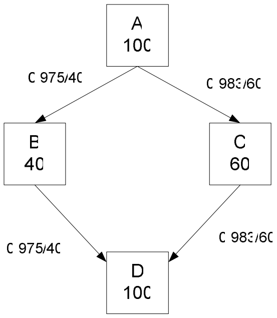


Figure 3.3: An example dependency graph

The dependency graph depicts only the dependency relations. It does not give any information about the type of dependency between the tasks i.e. whether they directly follow each other, or are parallel or alternate tasks. So, the next step is to determine the type of dependencies between the tasks.

3.2.1.2 Determining the dependency type

The dependency graph gives us the information about the input and output expressions of each task. For example, in Figure 3.3 the output of task A is a set of tasks B and C. The input to task D is also the same set of tasks. To discover the type of dependency between these tasks, the formula of AND measure in combination with a threshold (cf. Equation 3.2) is used to determine if the tasks are parallel to each other.

$$a \Rightarrow_w b \wedge c = \left(\frac{|b >_w c| + |c >_w b|}{|a >_w b| + |a >_w c| + k} \right)$$

Equation 3.2: AND measure

In general, the formula depicts a log W and three activities *a*, *b* and *c*. The $|a >_W b| + |a >_W c|$ indicates the number of times *a* is directly followed by *b* and *c* respectively and $|b >_W c| + |c >_W b|$ indicates the number of times *b* and *c* appear directly after each other. If Equation 3.2 gives a value equal to 1 then *b* and *c* are parallel activities i.e. they are in an AND relation. But if this value comes out to be 0.1 or less than 0.1 they are in an XOR-relation.

3.2.1.3 Discovering long distance dependencies

Sometimes in an event log two activities may not share a direct dependency relation but they may be related indirectly via some other activities. For instance, the log may always contain traces like ACBD, ACDBE, AECDB etc. it is apparent that the tasks A, B do not share a direct dependency but they are related to one another via some other tasks in between. This represents the non-free choice construct or non-local behaviour. To understand this, let us take an example shown in the Figure 3.4.

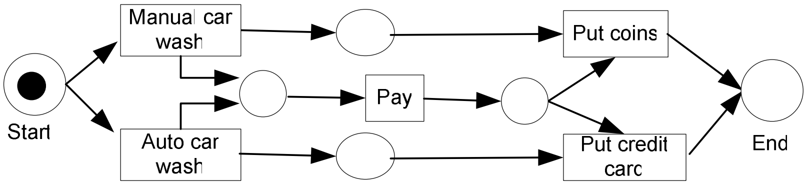


Figure 3.4: Non-free choice construct

The figure represents a Petri Net for car-wash. A person can wash his car by himself or he can get it washed by the personnel at the car wash station (automatic car wash). When he decides to wash his car by himself he is supposed to put coins in the machine available at the car wash station. In the second case, he is supposed to pay by credit card. He cannot pay by coins if he has not chosen to wash his car manually. It means the decision how to pay depends on choice made by the person earlier. This represents non-local behaviour that depends on some earlier choices made at some part of the process. Such constructs are difficult to mine in a process log as the choice is non local and the mining algorithm has to remember earlier events.

In this example a long distance dependency can be seen between the tasks ‘Manual car wash’, ‘Put coins’, and also between the tasks ‘Auto car wash’, ‘Put credit card’. The HM algorithm represents such long distance dependencies between activities *a*, *b* by the notation $a >>>_w b$. The formula for calculating $a >>>_w b$ is given in the Equation 3.3, where |*a*>>>w *b*| is the number of times *a*>>>*b* occurs.

$$a =>^{l}{}_{w} b = \left(\frac{2|a >>>_w b|}{|a|+|b|+1} \right) - \left(\frac{2Abs(|a|-|b|)}{|a|+|b|+1} \right)$$

Equation 3.3: Long distance dependency measure

Only after the long distance dependency relations are discovered a complete process model is obtained. This completes the construction of a process model using the HM algorithm.

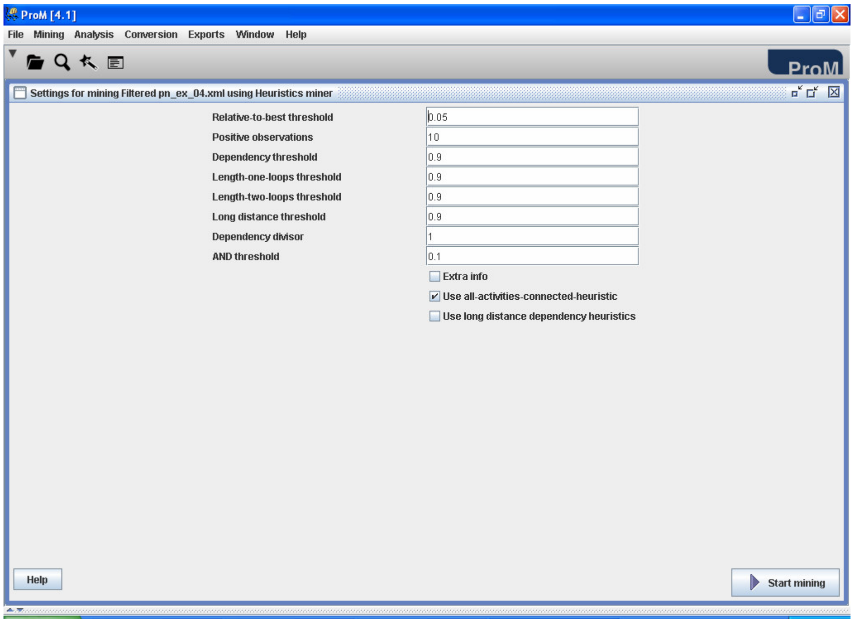
3.3 The Heuristics Mining plug-in

In this section, we introduce the Heuristics Mining plug-in implemented in the ProM framework. We first explain its parameters and then apply it to real-life processes pertaining to the healthcare domain.

3.3.1 The parameters

The discovery of a process model underlying an event log with the help of the HM is based on some parameters. Different values given to these parameters produce a different output, which can be analyzed to obtain meaningful conclusions. Figure 3.5 shows these parameters and their default values. The different parameters available in the HM are:

- All-events-connected-heuristic
- Dependency Threshold
- Dependency divisor
- AND Threshold
- Positive observations
- Relative-to-best Threshold
- Length-one-loops Threshold
- Length-two-loops Threshold
- Long distance threshold
- Long distance dependency heuristics
- Extra information



Settings for mining Filtered pn_ex_04.xml using Heuristics miner

Relative-to-best threshold	0.05
Positive observations	10
Dependency threshold	0.9
Length-one-loops threshold	0.9
Length-two-loops threshold	0.9
Long distance threshold	0.9
Dependency divisor	1
AND threshold	0.1

☐ Extra info
☑ Use all-activities-connected-heuristic
☐ Use long distance dependency heuristics

Figure 3.5: Parameters of the HeuristicsMiner algorithm

3.3.1.1 Use all-events-connected heuristic

We already discussed the *all-events-connected heuristic* in Section 3.2.1.1. In the Heuristics Mining plug-in if this parameter is selected, then a dependency graph is obtained on the basis of the fact that each non-initial activity must have at least one other activity that is its reason for execution and each non-final activity must have at least one activity that depends on it for its execution. This heuristic is used during the generation of a dependency graph based on the dependency values. When this parameter is used it ignores all other parameters.

3.3.1.2 Dependency threshold

The parameter *dependency threshold* represents a measure which is an indication of how sure we are of a dependency relation. Using the dependency threshold means that we will accept all those dependency relations from the event log whose value of dependency measure is equal to or greater than the value of the dependency threshold. As seen in Figure 3.5, the default value of the dependency threshold is 0.9. If the option for selecting the *all-events-connected heuristic* is selected then it overrides the use of parameter called '*dependency threshold'*.

Referring to the dependency graph in Figure 3.3, the dependency measure between tasks A and B is 0.975 and between tasks A and C is 0.983. Both these high values show that we are quite sure of these connections. But what is a good high value? It can be a relative decision therefore the parameter dependency threshold is defined. The dependency measure given by the formula in Equation 3.1 is used to generate dependency measures for all possible task/activity combinations available from an event log. Then all these values are compared with the specified dependency threshold parameter and accordingly accepted or rejected. If the calculated value is less than the specified value then the dependency between tasks *a* and *b* cannot be accepted for the generation of the process model. But, if the calculated value is equal to or higher than the specified value the dependency relation between *a*, *b* is accepted.

What if the dependency threshold is not used in the algorithm? This is an important yet simple question to answer. For example, we calculate dependency measure between two pairs of activities *a*, *b* and *p*, *q*. Suppose the dependency value for *a*, *b* comes out to be 0.95 and the value for *p*, *q* comes out to be 0.75. We may thus give an argument that since the value for the former pair is higher than the latter pair, so we should not trust the dependency relation between the latter pair and reject this relation. But how do we know how much high should a good high value be? Moreover, this high value is always affected by the presence of incorrect data, parallelism and the number of times an activity appears in the log. This dilemma is resolved by having a threshold value to compare the calculated dependency values with and make an acceptance/rejection decision based on this comparison.

3.3.1.3 Dependency divisor

The formula for dependency measure in Equation 3.1 has also hidden in itself yet another parameter for the HM algorithm. The denominator of the formula comprises of values: $| a >_w b |$, $| b >_w a |$ and k. This 'k' is the parameter: *Dependency Divisor*. The dependency divisor has a default value of 1 because 1 is a small number that can affect small logs (logs containing few traces) in a significant way and at the same time it has a less significant effect on the big logs. To understand the importance of dependency divisor, we consider two examples for a log. We vary the number of traces and the value of dependency divisor for this log. We also assume that *a*, *b* are any two activities in this log and only the activity *a* is directly followed by activity *b* and the other way round does not happen in the log.

Example1: Suppose a log W1 contains 5 traces with *a->b*. Now, let us see the effect on the values of dependency measure between *a*, *b* when we change the value of dependency divisor. Following table (Table 3.2) shows this.

Table 3.2: Number of traces with *a->b* =5

Dependency divisor	Dependency measure between *a*, *b*
1	0.83
2	0.71
5	0.2

Example 2: Now suppose W1 contains 50 traces with *a->b*. Now, let us see the effect on the values of dependency measure between *a*, *b* when we change the value of dependency divisor. Following table (Table 3.3) shows this.

Table 3.3: Number of traces with *a->b*=50

Dependency divisor	Dependency measure between *a*, *b*
1	0.90
2	0.88
5	0.84

It is very much apparent that the dependency measure between *a*, *b* is changed with a change in the value of dependency divisor. But what is of importance here is that we should understand whether the size of log and the changes in dependency divisor are inter-related. Tables 3.2 and 3.3 show that when the value of dependency divisor is changed in Example 1 then there are drastic changes in the dependency measures between *a*, *b*. For instance, when the dependency divisor =1, the value was 0.83 but when the former value is changed to 2 the dependency measure value becomes 0.71 and further becomes 0.2 in the third case. As opposed to these drastic changes, the changes in dependency values when dependency divisor is changed in Example 2 are very nominal. Therefore, it is clear that the changes in dependency divisor are more prominent in case of logs that contain lesser number of traces than in logs with a higher number of traces.

3.3.1.4 AND threshold

After the generation of dependency graph we need to know what kind of dependency exists between the activities represented in the dependency graph. This refers to discovering AND/XOR-split/joins. The Heuristics Mining plug-in provides this functionality in the parameter *AND threshold.* As the name suggests, this parameter indicates that two activities in a log are in parallel if their calculated AND measure (cf. Equation 3.2) value is greater than the specified value for the AND threshold. The default value of AND threshold is 0.1. When a dependency graph provides the semantics of splits/joins it is referred to as a Heuristics Net.

3.3.1.5 Positive Observations

The parameter *Positive Observations* enforces that the algorithm accepts only those dependency relations whose frequency is higher than the value of the Positive Observations threshold. It helps us to filter out low frequent patterns in the log and enables us to focus on the main behaviour of the log. A high value assigned to this parameter indicates a user's interest in high reliability of the fact that an activity is directly followed by another activity. To understand this concept, please refer to Appendix D.

3.3.1.6 Relative-to-best threshold

The *Relative-to-best threshold* indicates that we will accept a dependency measure for which the difference with the "best" dependency measure is lower than the value of relative-to-best threshold. A high value of Relative-to-best threshold shall generate detailed behaviour as then the model would also include dependency relations with low dependency values. To understand the concept behind this parameter readers are referred to Appendix D where it is explained with the help of some examples.

3.3.1.7 Length-one-loops threshold

In an organization it is quite possible that an activity is repeated multiple times. For instance, in a travel agency an employee tries to contact a customer. If he is not able to contact him the first time, he makes another attempt. In this case, the activity 'contact customer' is repeated several times depending on the situation. Figure 3.6 represents this repetition:

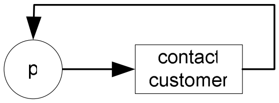


Figure 3.6: Length-one-loop example

In Petri net terminology, when a transition consumes its own token it is referred to as length-one loop (L1L). Because the loop involves only one activity, it is called length-one-loop. The HM deals with L1Ls through the parameter *length-one-loops threshold*. The value of the parameter can be set to discover length-one-loops in an event log. The default value is 0.9. The formula for dependency measure showing the dependency value for an activity *a* following itself is given below as Equation 3.4. It should be noted that $|a >_w a|$ is the number of times the activity *a* directly follows itself. The traces that are observed in event logs for length-one-loops are of the form "…AA…."

$$a \Rightarrow_w a = \left(\frac{|a >_w a|}{|a >_w a| + 1} \right)$$

Equation 3.4: Length-one-loop dependency measure

A L1L is accepted in a process model if the calculated value from the above equation is higher than the specified threshold value. If we assign a lower value to this parameter we are able to discover loops that are low frequent. The L1L threshold thus provides a way to capture low frequent behaviour of loops found in the log.

3.3.1.8 Length-two-loops threshold

Length-two-loops (L2L) are the loops between two activities as shown in the figure below. In such a situation we would expect the traces of the form "…ABAB…" etc.

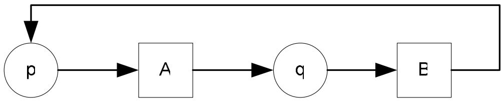


Figure 3.7: Length-two-loop example

In a process model such loops have only one cause and one depending activity. The HM has to deal with them with proper attention because these loops may be misunderstood for series transitions of the form AB and BA instead of a loop with trace ABAB. In the former case, it may add up to the number of wrong observations of every occurrence of BA as in the log $a>_w b$ as well as $b>_w a$, both will occur. Thus, a different formula for L2L dependency value is needed. This formula for calculating the L2L dependency measure between two activities *a*, *b* is given below:

$$a \Rightarrow_{2\ w} b = \left(\frac{|a >>_w b| + |b >>_w a|}{|a >>_w b| + |b >>_w a| + 1} \right)$$

Equation 3.5: Length-two-loop dependency measure

In Equation 3.5, |*a*>>w *b*| is the number of traces of the form “aba”, and |*b*>>w *a*| is the number of traces of the form “bab”. The formula represents that we will accept dependency relations between activities in L2L that has a dependency value higher than or equal to the value of L2L threshold. A lower value to this parameter discovers low frequent length-two loops. The default value is 0.9.

3.3.1.9 Long distance threshold & Use long distance dependency heuristics

As already introduced in Section 3.2.1.3 some choices are controlled in some other part of the process model, far from where actually the effect of the choice is realized. This non-local behaviour is captured by the *long distance dependency heuristic.* This parameter indicates to the algorithm that we are also interested in those dependencies which are not only indirect but long distance in nature. Figure 3.8 shows a heuristics net generated with the option to discover long distance dependencies. A long distance dependency exists between activities *B*, *E* (activity *B* is in an AND split with activities *D* and *E*, the direct arc from it to the activity *E* is a long distance dependency). The value of the parameter *long distance threshold* specifies which long distance dependencies to accept/reject. If the value of the long distance dependency measure (cf. Equation 3.3) is less than the value of long distance threshold, the dependency will be rejected. The default value of this threshold is 0.9.

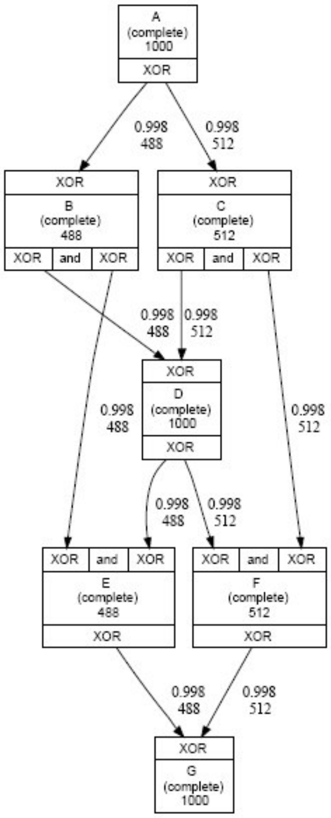


Figure 3.8: Long distance dependency

3.3.1.10 Extra information

The Heuristics Mining plug-in also generates some additional mining information. This information helps us to understand how and why a particular output is generated. Readers can refer to Appendix D to take a look at this information generated for the event log used to mine the model in Figure 3.8

3.3.2 Experimenting with the HeuristicsMiner algorithm

In this section, experiments with the HM algorithm using real-life healthcare processes from Case study1 are illustrated. The algorithm is used to generate process models for these processes. Through these experiments the focus is on the second research goal:

Analyze the process models derived from the HeuristicsMiner algorithm and evaluate them on criteria such as simplicity, ease of understanding and the degree of meaningful information obtained from them.

Till now the HM algorithm was tested using benchmark artificial data in [11] and it was found that the algorithm is till date one of the most robust algorithms for event logs containing noise. Therefore we would also like to determine if the performance of the algorithm on real-life logs is similar to its performance on the benchmark material. These experiments on this material were done with the default parameter values so the effect of different parameter settings on the output of the algorithm will also be studied in our experiments. For all our experiments database tables converted to MXML logs will be used. In Appendix E readers can take a look at these logs along with the tables that were used for forming these logs.

Before describing the experiments conducted in context of the second research goal of this thesis, we would first like to explain a simple experiment with the Heuristics Mining plug-in in order to understand its output. We take a log consisting of 5000 cases and 13 different ATEs. The resulting heuristics net after applying the algorithm with default parameter settings (relative-to-best threshold=0.05, positive observations=10, dependency threshold=0.9, Length-one-loops threshold=0.9, Length-two-loops threshold=0.9, Long distance threshold=0.9, dependency divisor=1, AND threshold=0.1, use-all-events-connected heuristic=true and use long-distance dependency heuristics=false) is shown in Figure 3.9.

The heuristics net provides us the information about: the tasks in the process log, frequencies of these tasks and their dependencies, dependency measures of these tasks and the split/join semantics. For instance, in Figure 3.9 it is seen that i) *a* is the start activity, *x* is the end activity and both of them happens 5000 times as indicated in their respective activity box, ii) the dependency measure of a->c is 1, iii) activity *c* is an XOR split with activities *g* and *d*. The plug-in also gives this split/join semantics in text. These semantics are seen in Figure 3.10.

Let us briefly understand these semantics by taking a part of this information and analyzing it. For example, we take a look at the details of the activity *g (complete)*. The IN [ ] represents the ingoing connections to this activity and the OUT [ ] represents its outgoing connections.

```
Element "g (complete)":
In: [ [ "c (complete)" "d (complete)" ] ]
Out: [ [ "i (complete)" "h (complete)" ] ]
```

In: [["c (complete)" "d (complete)"]], indicates that activities *c* and *d* are in an XOR relation and forms an XOR join at the activity *g*. Similarly, Out: [["i (complete)" "h (complete)"]] indicates that activities *i* and *h* are in XOR relation and forms an XOR split from the activity *g*. Here, these activities are in an XOR relation as they are in the same subset. But if we had: OUT:[[“c (complete)”] [“d (complete)”]], this would mean the activities *c* and *d* are in an AND relation as they belong to two different subsets.

Besides the information conveyed by the heuristics net, following textual information is also obtained:

- Number of process instances: It shows the number of cases recorded in this event log.
- Number of audit trail entries: It is the number of tasks in the log (along with event type information).

- Total number of connections: This information pertains to the number of arcs in the Heuristics Net.
- Total wrong observations: Wrong observations indicate noise. The number of wrong observations for this process model is 0 indicating that the mined event log is free from any sort of noise.
- Fitness measure: Fitness is a measure that gives an indication of the extent to which the log traces comply with the generated process model. It measures the distance between the behaviour actually observed in the log and the behaviour described by the process model. We can see *continuous semantics fitness* and *improved continuous semantics fitness*. The fitness of the process model is 1, indicating that all the log traces are successfully parsed. The reader is referred to Appendix G to understand the concepts of fitness.

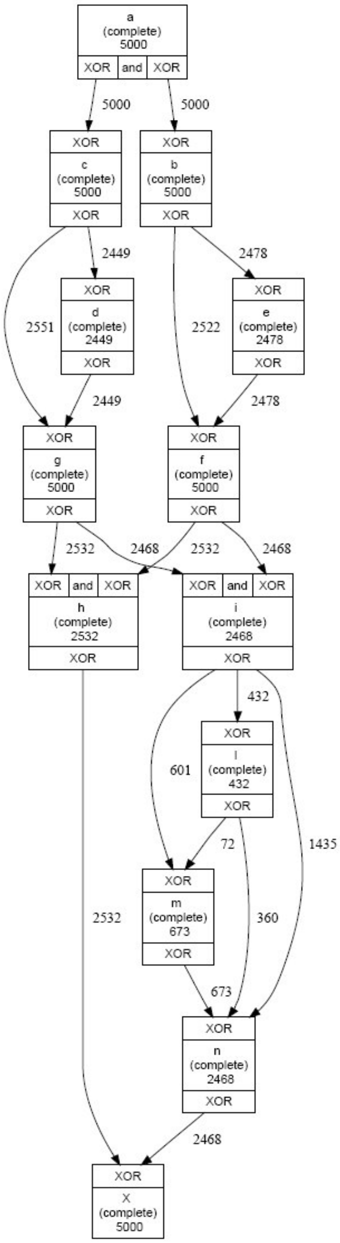
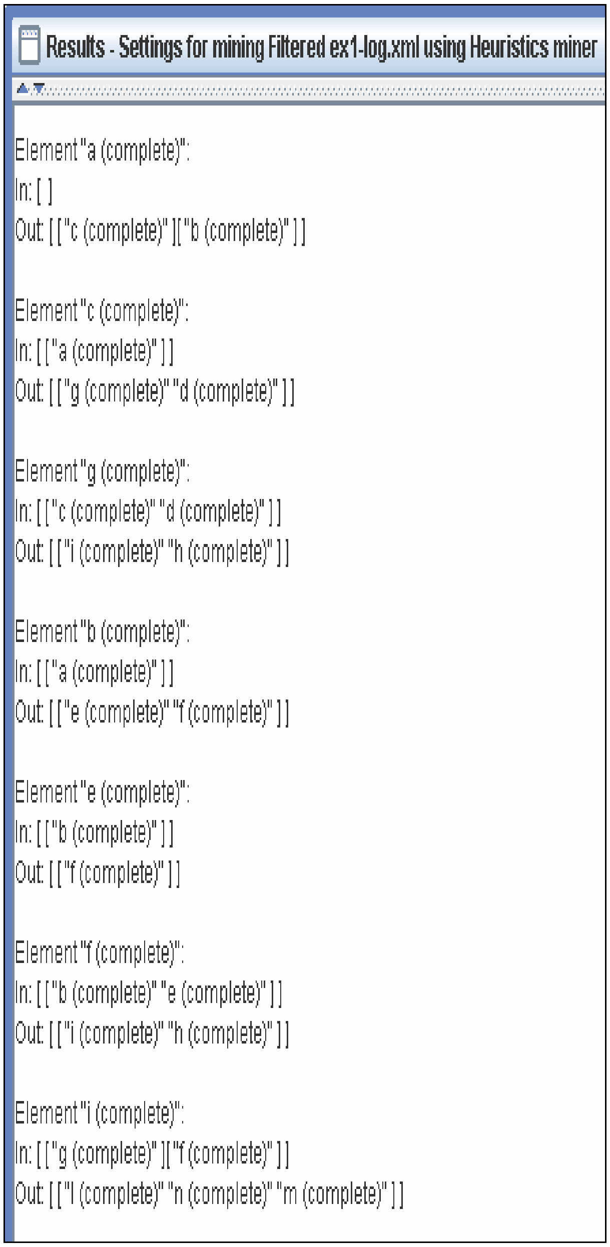

a (complete) 5000
XOR and XOR
5000
5000
XOR c (complete) 5000 XOR
XOR b (complete) 5000 XOR
2449
2478
2551
2522
XOR d (complete) 2449 XOR
XOR e (complete) 2478 XOR
2449
2478
XOR g (complete) 5000 XOR
XOR f (complete) 5000 XOR
2532
2468
2532
2468
XOR and XOR h (complete) 2532 XOR
XOR and XOR i (complete) 2468 XOR
432
601
XOR l (complete) 432 XOR
72
1435
XOR m (complete) 673 XOR
2532
360
673
XOR n (complete) 2468 XOR
2468
XOR X (complete) 5000

Figure 3.9: An example heuristics net



Results - Settings for mining Filtered ex1-log.xml using Heuristics miner

```
Element "a (complete)":
In: [ ]
Out: [ [ "c (complete)" ] [ "b (complete)" ] ]

Element "c (complete)":
In: [ [ "a (complete)" ] ]
Out: [ [ "g (complete)" "d (complete)" ] ]

Element "g (complete)":
In: [ [ "c (complete)" "d (complete)" ] ]
Out: [ [ "i (complete)" "h (complete)" ] ]

Element "b (complete)":
In: [ [ "a (complete)" ] ]
Out: [ [ "e (complete)" "f (complete)" ] ]

Element "e (complete)":
In: [ [ "b (complete)" ] ]
Out: [ [ "f (complete)" ] ]

Element "f (complete)":
In: [ [ "b (complete)" "e (complete)" ] ]
Out: [ [ "i (complete)" "h (complete)" ] ]

Element "i (complete)":
In: [ [ "g (complete)" ] [ "f (complete)" ] ]
Out: [ [ "l (complete)" "n (complete)" "m (complete)" ] ]
```

Figure 3.10: Splits and Joins information

After explaining the various parts of the output of the Heuristics Mining plug-in, the next section illustrates the experiments conducted with the plug-in.

3.4 Experiments with the HeuristicsMiner

As already mentioned, the purpose of the following experiments is to get insights into the healthcare processes by analyzing their process models generated by the HM algorithm. We describe our experiments under the heading *Illustration* and each illustration contains experiments conducted with a specific purpose. Illustration 1 describes the output of the HM algorithm and discusses the problems with it. Illustration 2 and 3 describes experiments performed on logs obtained after applying some filtering mechanisms. These experiments also show the effect of different parameter settings on the output of the HM.

3.4.1. Illustration 1

The healthcare log used in this experiment pertains to different complications patients suffer from. This log has 576 process instances (PIs) and 185 different ATEs. These PIs represent different 'complication paths' followed by different patients i.e. these PIs show for different patients the order in which their one complication leads to another. Each event in a PI is a complication, e.g., *C_Febris e.c.i, C_Anurie (<1ml/kg/24u), C_Aspiratie*, etc.

The algorithm was applied with default parameter settings and the output of this experiment is shown in Figure 3.11. As seen in this figure, the screen is divided into two parts by a separator bar. On the right hand side (RHS) the structure of the complete process model can be seen and the left hand side (LHS) shows a part of this complete process model. Following observations were made from this process model:

- The process model has a complex spaghetti-like structure.
- Presence of dangling[11] activities like *C_Abces (start)*, *C_N Phrenicus Paralyse (start)* etc.
- Besides being complex, the process model misses many dependencies. For instance, it can be seen from the dependency graph that the activity *C_Febris e.c.i (start)* is registered in the log 6 times but the process model captures only two of its outgoing connections. The remaining 4 connections are missing. Similarly, the process model does not capture all connections for other activities like *C_Hypoglycaemie (complete)*, *C_Pleura-Effusie (start)* etc.
- The Improved Continuous semantics fitness of the model is -0.44.

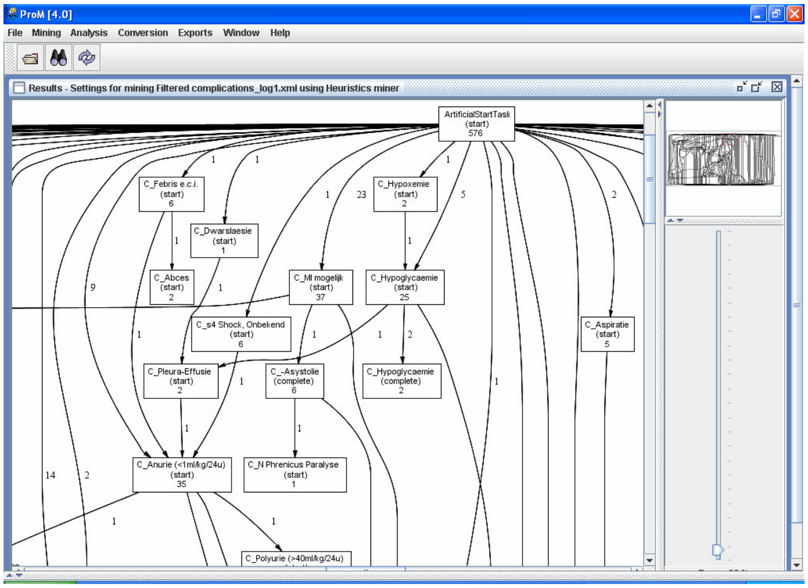

ProM [4.0]
File Mining Analysis Conversion Exports Window Help
Results - Settings for mining Filtered complications_log1.xml using Heuristics miner
ArtificialStartTask (start) 576
C_Febris e.c.i. (start) 6
C_Hypoxemie (start) 2
C_Dwarslaesie (start) 1
C_Abces (start) 2
C_MI mogelijk (start) 37
C_Hypoglycaemie (start) 25
C_s4 Shock, Onbekend (start) 6
C_Aspiratie (start) 5
C_Pleura-Effusie (start) 2
C_-Asystolie (complete) 6
C_Hypoglycaemie (complete) 2
C_Anurie (<1ml/kg/24u) (start) 35
C_N Phrenicus Paralyse (start) 1
C_Polyurie (>40ml/kg/24u)
1
1
1
23
5
2
9
1
1
1
1
1
2
1
1
1
1
14
2
1
1

Figure 3.11: Process model for complications log, fitness= -0.44

Before analyzing these observations, we would also like to mention the experiment with another healthcare log. The log for this experiment consists of information about treatments given to various patients. It has

[11] A Dangling activity is an activity with no dedicated start and/or end points in a process model.

2711 PIs and 253 different ATEs. These PIs represent different ‘treatment paths’ followed by different patients i.e. these PIs show for different patients the order in which their one treatment is followed by another. Each event in the PI is a treatment, e.g., *B_Isolatie_druppel, B_Isolatie_strikte, B_Halsinf/subclavia op IC, B_Halsinf/subclavia op OK*, etc. The output of the algorithm applied with default parameter settings is shown in Figure 3.12. The Continuous Improved semantics fitness of the model is -0.64. Other observations that were made for this process model are similar to the one made for the first experiment with the complications log. From the Figure 3.12 it is apparent that the process model is very complex. Dangling activities and missing dependencies were also observed. Similar results were obtained for many other event logs from Case study1.

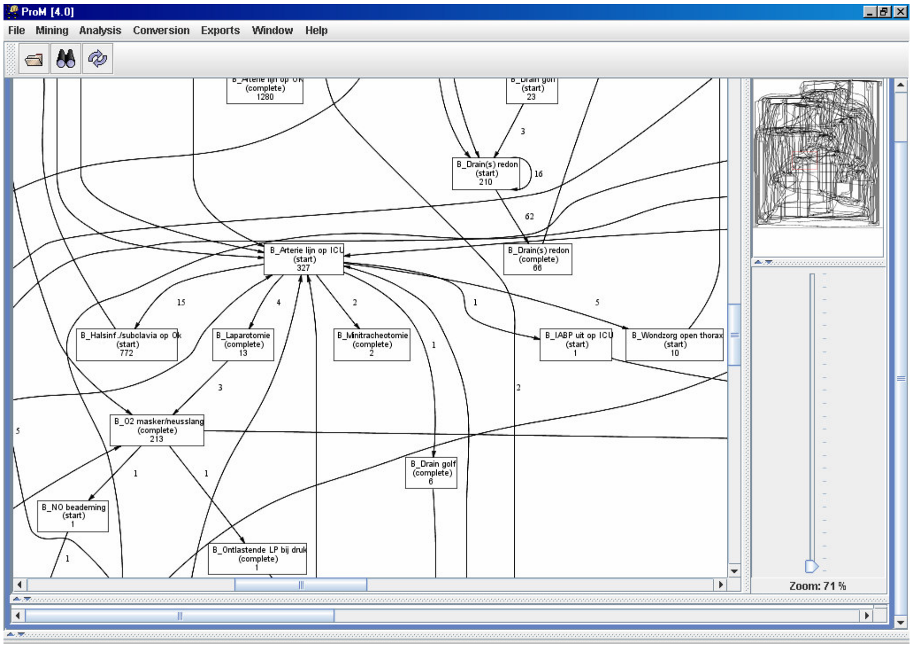

ProM [4.0]
File
Mining
Analysis
Conversion
Exports
Window
Help
B_Arterie lijn op OK (complete) 1280
B_Drain golf (start) 23
3
B_Drain(s) redon (start) 210
16
62
B_Arterie lijn op ICU (start) 327
B_Drain(s) redon (complete) 66
15
4
2
1
5
B_Halsinf./subclavia op Ok (start) 772
B_Laparotomie (complete) 13
B_Minitracheotomie (complete) 2
1
B_IABP uit op ICU (start) 1
B_Wondzorg open thorax (start) 10
3
2
B_02 masker/neusslang (complete) 213
5
1
1
B_Drain golf (complete) 6
B_NO beademing (start) 1
1
B_Ontlastende LP bij druk (complete) 1
Zoom: 71 %

Figure 3.12: Process model for treatments log, fitness= -0.64

Based on the characteristics of the healthcare described in Section 1.1.1, observations from the Case study1 and the understanding of the HM algorithm, we attempt to analyze the observations made for these two experiment logs:

- The complexity of the process models can be attributed to the *uniqueness of cases i.e. patients* in the healthcare domain. Each patient represents a unique and distinct case depending on his specific conditions like previous medical history, responses to certain drugs/treatments/complications and various other factors. It is difficult to say that for example, 10 patients suffering from a complication say *A* always suffers from the same complications thereafter. Depending on the specific condition of a patient, he may suffer from different complications following the complication *A* and hence receive different treatments. This illustrates heterogeneity of patients in the healthcare domain. Owing to this **heterogeneity and uniqueness of patients** present in the event log, the HM algorithm results into a complex process model.
- In spite of the addition of the unique start and end events[12], some dangling activities are found in the process model. The reason can be the ***presence of noise***. It is quite possible that due to registration errors some unwanted events are inserted or some relevant events are missed from the event log. This may result into missing further connections between any dangling activities with other activities in the log. This shows some possible problem/issue from the algorithm point of view.

[12] Addition of Artificial Start and End events in an event log.

- Some missing connections are also observed in the process model. For example, an activity though registered 10 times in the log, the process model captures only 4 of its outgoing connections. The remaining 6 connections for this activity are not found in the model. This can again be the result of **noise** in the log. It is possible that erroneous insertion/deletion occurred in the event log resulting into loss of relevant connections for some activities.
- Besides missing connections, the process model also misses some events which are registered in the log but are not captured in the model. This may be because they are low frequent events and these are left out in the model because they **do not fulfil certain parameters of the algorithm** (cf. Section 3.3.1).
- The **low frequent events** found in the log may be due to noise or medical exceptional cases. But it can also not be ignored that the low frequent events are common in the ICU[13] of any healthcare organization. At ICU severely ill patients are admitted and these patients may suffer from many problems. The course of treatment constantly needs to be determined for such patients. In this situation it is difficult to find any standardized process. Therefore, the HM algorithm does not generate simpler models for such flexible processes. But it is also equally true that the HM algorithm cannot distinguish low frequent behaviour from noise.
- Fitness is a quality measure indicating the gap between the behaviour actually observed in the log and the behaviour described by the process model. It gives the extent to which the log traces can be associated with execution paths specified by the process model [23]. Both of the above models have a poor fitness value (negative values) indicating that most of the log traces are not successfully parsed by the mined process model. This may be because of the presence of noise resulting into dangling activities and missing connections. It is also possible that the parameter settings do not discover all connections.

From this analysis we can say that the complexity of the process model, presence of dangling activities and other problems mainly stem from the underlying investigated healthcare process. We can make an attempt to obtain understandable process models by varying the parameter settings of the HM algorithm. In the next illustration, the impact of varying the parameter settings on the output of the HM is discussed.

3.4.2. Illustration 2

For the experiments described here, same logs as used in Illustration 1 are used. The process models obtained from different parameter settings are not shown due to the limitation of space, but the key observations are mentioned below:

- To generate the main behaviour of the process, we set high values for parameters like Positive Observations, Dependency threshold, Length-one-loops threshold and Length-two-loops threshold. As already indicated in Section 3.3.1, higher values of these parameters generate main behaviour of the process. The resulting process models obtained with these parameter settings were less complex as compared to models in Figure 3.11 and 3.12 but a lot of dangling activities and missing connections are still observed.
- When the event log for complications was mined with default parameter settings except using the *all-activities-connected heuristic*, totally disconnected activities were observed. In presence of the *ArtificialStartTask* and *ArtificialEndTask*, the activities are connected only to the start and end tasks, and a lot of dangling activities are also present. Besides, these connected activities, a lot of disconnected activities are also found. Readers can refer figures F.1 and F.2 in Appendix F to take a look at these two process models.
- When the event log for treatments was mined with default parameter settings except using the *all-activities-connected heuristic*, although a better model was obtained compared to models in Figure 3.11 and 3.12 (in terms of simplicity and ease of understanding) but a lot of dangling activities, totally unconnected activities and missing connections are also found. These models can be found in figures F.3 and F.4 in Appendix F. These models contain a lot of low frequent events. As already mentioned that these may be due to noise or actually low frequent event. So, if the log is cleaned from such low frequent events (which are also shown as dangling activities) and then the HM algorithm is applied, simpler and complete models may be obtained.

[13] Case study1 contains data from the ICU department of the Catharina hospital.

From the analysis in Illustration 1 and 2, we can say that the complexity of the process model, presence of dangling activities and other problems mainly stem from the underlying investigated healthcare process. The input to the algorithm is a dynamic, flexible and less structure event log. The HM does not simplify the output from the complex input it receives in form of these logs. If the parameter *all-events-connected heuristic* is not used then we get simpler models but the behaviour represented in this model is very much incomplete. The focus and emphasis in a domain like the healthcare is on the simplicity and the completeness of the model. Therefore, some techniques must be found out to retrieve simpler and easier to understand process models. One of the ways can be abstracting the input event log in order to retain only some desired portions of the log. Below we give a brief overview of how abstraction can be achieved.

Abstraction is a relative process. It highly depends on what results one would like to obtain. For example, from the complications log, the interrelationship between various complications may be of interest or it would be interesting to find complications found in patients of certain age group. An event log that records information about patient's complications and treatments would be interesting to discover what treatment procedures are followed for complications of a specific category. Based on the desired results an event log can be abstracted. In the ProM framework, abstraction can be done in the following ways:

- Using different filters: The ProM framework offers a variety of filters. For example, the *Event log filter* enables the selection of only desired activities, the *Enhanced event log filter* enables a user to specify relative percentage of an activity in the entire log whereas activities performed by a particular originator can be selected using the *Originator log filter* etc. Based on a user's requirements these filters can be used to abstract an event log. Further, the instances satisfying the filtering criteria can be then exported to a new log file using the Export plug-ins.
- Specify certain properties using the Linear Temporal Logic (LTL) analysis plug-in: The default LTL checker plug-in can be used to abstract on the basis of control flow, originators etc. For example, instances in which say, complication_1 is always/eventually followed by complication_2 can be retrieved and exported.
- Arc pruning: Some arcs in a mined model can be pruned (removed from the model) that are used fewer times than a certain threshold (that can be specified by the user). The threshold refers to the arc usage percentage which is relative to the most frequently used arc [16]. For example, the most frequent arc usage in the model is 100. If this threshold is set to 5%, all arcs of this model that are used 5 or fewer times are removed from the model. This way we can focus on the main behaviour of the process and the log can be filtered from certain low frequent behaviour.

It should however be noted that by abstraction a smaller part of the entire log is used and the behaviour that do not satisfy the criterion is lost. But it is equally true that a logically abstracted log makes it easy to focus on investigating the obtained process model underlying a healthcare process. It also becomes easy to investigate how different parameters of the algorithm interact to produce a certain output. The experiments illustrated now onwards are done on abstracted logs as we intend to provide the HM with simple input logs in order to achieve simple and nicer process models unlike the complex models retrieved in the above experiments.

3.4.3. Illustration 3

The experiments illustrated here were performed on healthcare logs abstracted on the basis of the category the complications belong to. The log that is used in this experiment stores information about patients suffering from the complications of the category 'uro-genitaal'. It contains 6 PIs and 18 different ATEs. We chose such a small log so that the impact of parameters on the output can be studied. The result of applying the HM algorithm (with default parameter settings) to this log is displayed in the Figure 3.13. A small and simpler process model as opposed to models in figures 3.11 and 3.12 is apparent in the Figure 3.13. Some general observations can be made from this model, for example, it can be inferred that a patient suffering from the complication *C_Trombopenie* typically also suffers from the complication *C_Oligurie* (and sometimes this complication may result into the complication *C_Bloeding waarvoer reOK)*. However, on carefully analyzing the model, the same problems as observed for models in Figure 3.11 and 3.12 were encountered. For instance, the activity 'ArtificialEndTask' is registered in the log 6 times but is actually captured only 5 times by the model. As the log is very small it was possible to cross check with the event log to find whether the process model is consistent with information recorded in the log. On this it was discovered that some connections are missing. For instance, the activity '*C_VKF, atrium flutter'* should connect to the activities 'ArtificialEndTask' and '*C_oligurie*' but these connections are missing in the process model. One of the L1L (activity '*C_Psychose/verward*') is also not captured in the model.

For the missing connections it was figured out that may be the value of their dependency measure is lower than the acceptable dependency threshold, or the number of times they are directly followed by some another activity is less than the value of positive observations. Based on this reasoning the experiment was re-performed with changed parameter settings (dependency threshold=0.5, positive observations=1, L1L threshold=0.5). The model obtained with the new settings is displayed in Figure 3.14. It is seen that now the missing connections in the model (cf. Figure 3.13) are captured. This happened because when the value of the dependency threshold is changed from 0.9 to 0.5 it was indicated to the algorithm to accept connections even with the dependency measure equal to 0.5. Lowering the value of positive observations resulted in the algorithm to produce those connections whose frequency (consistent with the definition of positive observations threshold, cf. Section 3.3.1.5) was higher than 1 (and not higher than 10 as with default value). The lowered value of L1L threshold indicated that loops with lower dependency measure would also be accepted. These changes in the parameter settings eliminated the problems found in the model with default parameter settings. Also, the fitness value improved from 0.88 to 1 indicating that all the log traces are correctly parsed. This model (cf. Figure 3.14) gives insights into the control flow of patients suffering from complications of type 'uro-genitaal'.

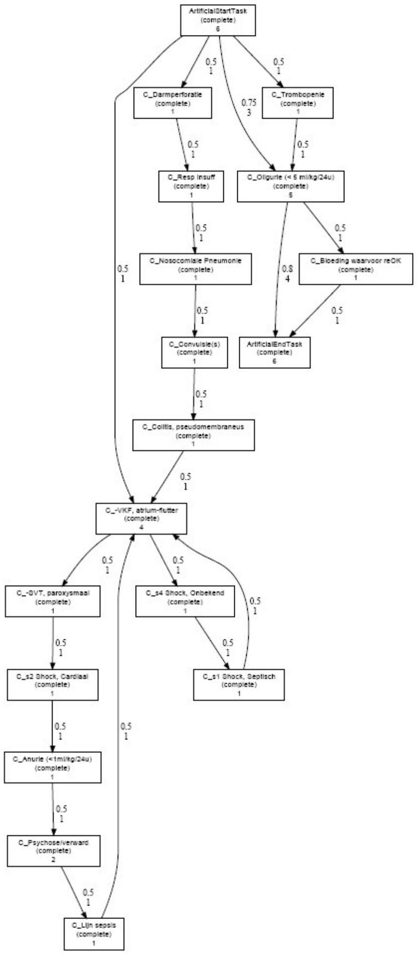

ArtificialStartTask (complete) 6
C_Darmperforatie (complete) 1
C_Trombopenie (complete) 1
C_Resp Insuff (complete) 1
C_Oligurie (< 5 ml/kg/24u) (complete) 5
C_Nosocomiale Pneumonie (complete) 1
C_Bloeding waarvoor reOK (complete) 1
C_Convulsie(s) (complete) 1
ArtificialEndTask (complete) 6
C_Colitis, pseudomembraneus (complete) 1
C_-VKF, atrium-flutter (complete) 4
C_-SVT, paroxysmaal (complete) 1
C_s4 Shock, Onbekend (complete) 1
C_s2 Shock, Cardiaal (complete) 1
C_s1 Shock, Septisch (complete) 1
C_Anurie (<1ml/kg/24u) (complete) 1
C_Psychose/verward (complete) 2
C_Lijn sepsis (complete) 1
0.5 1
0.75 3
0.8 4

Figure 3.13: Process model mined with default parameter settings.

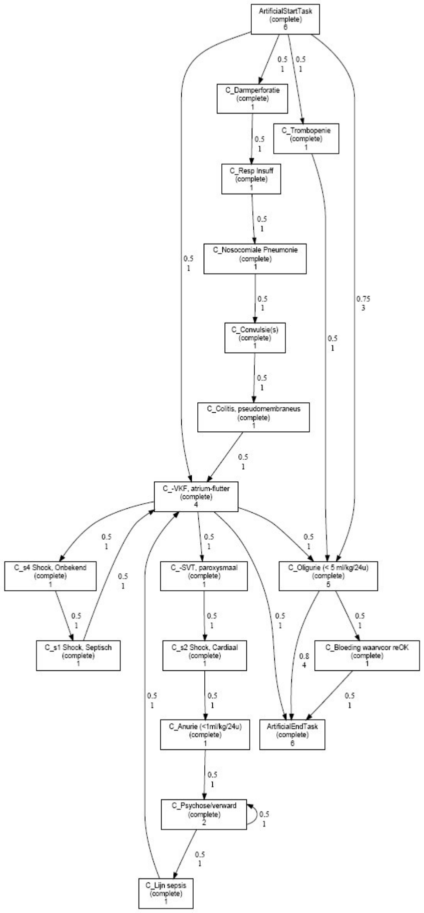

ArtificialStartTask (complete) 6
C_Darmperforatie (complete) 1
C_Trombopenie (complete) 1
C_Resp Insuff (complete) 1
C_Nosocomiale Pneumonie (complete) 1
C_Convulsie(s) (complete) 1
C_Colitis, pseudomembraneus (complete) 1
C_-VKF, atrium-flutter (complete) 4
C_s4 Shock, Onbekend (complete) 1
C_-SVT, paroxysmaal (complete) 1
C_Oligurie (< 5 ml/kg/24u) (complete) 5
C_s1 Shock, Septisch (complete) 1
C_s2 Shock, Cardiaal (complete) 1
C_Bloeding waarvoor reOK (complete) 1
C_Anurie (<1ml/kg/24u) (complete) 1
ArtificialEndTask (complete) 6
C_Psychose/verward (complete) 2
C_Lijn sepsis (complete) 1
0.5 1
0.75 3
0.8 4

Figure 3.14: Change in parameter settings can discover the desired clean model.

Besides giving insight into the underlying process, this experiment also illustrated how parameter settings can affect the output of the plug-in. Several experiments were performed to generalize the fact that changes in the parameter settings can discover the desired and clean process model (free from problems encountered in experiments till now), but it was not found to be true for every event log. The experiment mentioned below illustrates that the parameters cannot always help in obtaining the desired model.

The experiment illustrated here was performed on a healthcare log abstracted on the basis of the complication category. It stores information about patients suffering from the complications of the category 'CNS' involving problems with the respiration system. This log consists of 15 PIs and 22 different ATEs. The result of mining with default parameter settings is shown in Figure 3.15. The analysis of the process model led to the discovery of problems like: missing dependencies, undiscovered loops, dangling activity (the activity *C_Empyeem*), poor fitness (-0.17) etc.

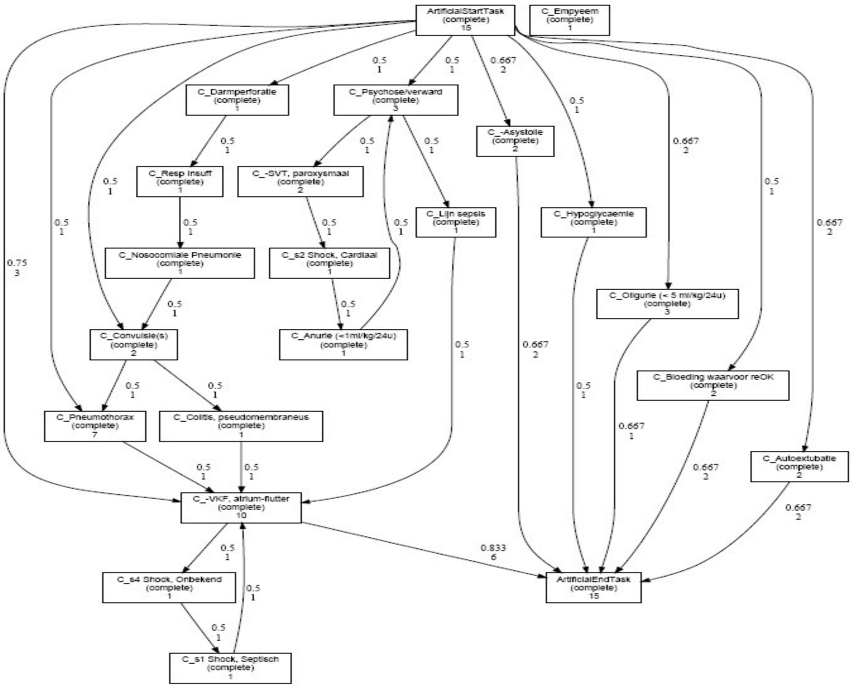

ArtificialStartTask (complete) 15
C_Empyeem (complete) 1
0.5 1
0.5 1
0.667 2
C_Darmperforatie (complete) 1
C_Psychose/verward (complete) 3
0.5 1
C_-Asystolie (complete) 2
0.5 1
0.5 1
0.5 1
0.667 2
C_Resp Insuff (complete) 1
C_-SVT, paroxysmaal (complete) 2
0.5 1
0.5 1
C_Lijn sepsis (complete) 1
C_Hypoglycaemie (complete) 1
0.5 1
0.75 3
0.5 1
0.5 1
0.5 1
0.667 2
C_Nosocomiale Pneumonie (complete) 1
C_s2 Shock, Cardiaal (complete) 1
C_Oligurie (< 5 ml/kg/24u) (complete) 3
0.5 1
0.5 1
C_Convulsie(s) (complete) 2
C_Anurie (<1ml/kg/24u) (complete) 1
0.5 1
0.667 2
0.5 1
0.5 1
C_Bloeding waarvoor reOK (complete) 2
C_Pneumothorax (complete) 7
C_Colitis, pseudomembraneus (complete) 1
0.667 1
0.5 1
0.5 1
0.667 2
C_Autoextubatie (complete) 2
0.667 2
C_-VKF, atrium-flutter (complete) 10
0.833 6
0.5 1
C_s4 Shock, Onbekend (complete) 1
0.5 1
ArtificialEndTask (complete) 15
0.5 1
C_s1 Shock, Septisch (complete) 1

Figure 3.15: Process model for complication category: CNS (mined with default parameter settings)

As illustrated for the model in Figure 3.13 a change in parameter settings may enable the discovery of a process model free from such problems. Therefore this experiment was performed with several different parameter settings. It was noticed that in this process of changing parameters, the process model undergoes several changes, sometimes good, and sometimes bad. Some parameter settings discover some missing connections and at the same time create dangling activities. It was seen that in spite of experimenting with the parameter values, inconsistencies in the process model remain and the various process models obtained with different parameter settings illustrate the huge dependency of the model on the parameter settings. In this case it is difficult to reach to one optimum parameter setting which can discover a clean and problem free process model.

During experiments with different healthcare logs from Case study1 we also encountered a problem apart from the problems mentioned till now. The next Illustration describes this new problem.

3.4.4. Illustration 4

For this illustration we experiment with a treatment log that has been filtered to retain the activities occurring more than a specified percentage (we used the Enhanced Log filter available in ProM, and set percentage task =6.03% and percentage PI=0) in the event log. This log consists of 2711 PIs representing different patients undergoing various treatments and 9 different ATEs representing 9 different treatments these patients receive. Figure 3.16 shows the process model mined with the default parameter settings of the HM.

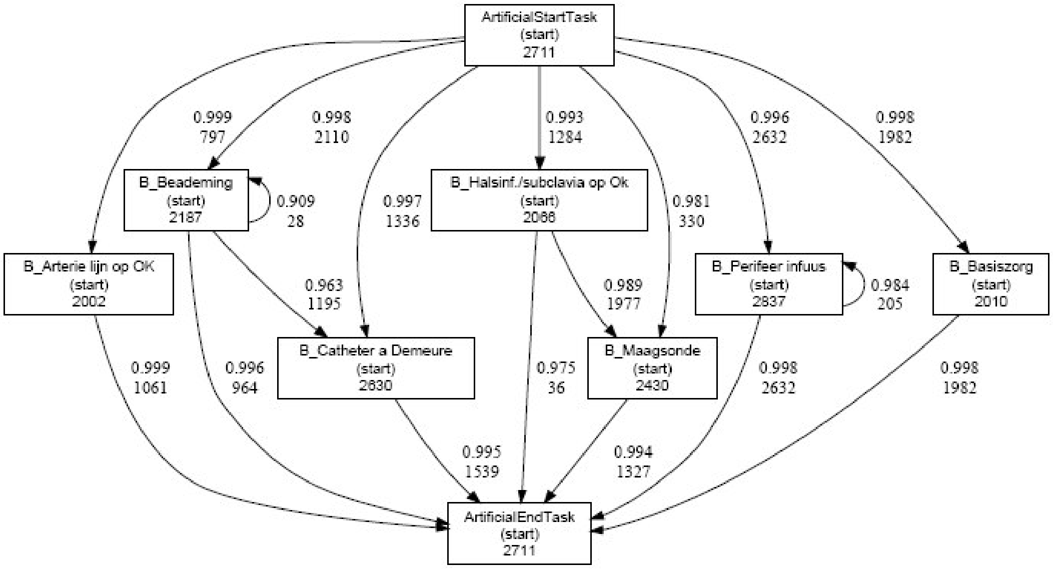


Figure 3.16: Process model for a treatment log with activities occurring more than a specified percentage

The Figure 3.16 depicts some general information like length-one-loops and which treatment activities are followed by other activities in this event log. It can also be seen that a lot of connections are missing. For instance, the activity *B_Cathether a Demeure* is executed 2630 times (according to the event log) but the label on the arc shows 1539 indicating that rest of its connections are missing in the process model. This is true for other activities too. It should be noted that this cannot be attributed to the low frequent behaviour which is not captured by the algorithm because the activities are quite frequent.

Other information derived from the model is about the join and split semantics. From the semantics information it can be seen that the activities within one pair of [ ] are in an XOR relation with one another and such pairs are in an AND relation with other pairs. The following semantics information was obtained:

Element "ArtificialStartTask (start)":
In: [ ]
Out: [["B_Beademing (start)"]["B_Perifeer infuus (start)"]["B_Basiszorg (start)"]["B_Arterie lijn op OK (start)" "B_Catheter a Demeure (start)"]["B_Arterie lijn op OK (start)" "B_Halsinf./subclavia op Ok (start)" "B_Maagsonde (start)"]]

It is seen that the set of activities- ["B_Arterie lijn op OK (start)" "B_Catheter a Demeure (start)"] are in an XOR relation, and therefore the sum of their frequencies ideally should be equal to the frequency of their outgoing task i.e. the ArtificalEndTask in this case. But this is not the case. None of the [ ] pairs fulfil this property. This is due to the missing connections. So, in order to discover the missing connections and thus obtain a more complete model different parameter settings were tried. For the parameter settings: relative-to-best threshold =0.99, positive observations = 1, dependency threshold =0.1, length-one-loops-threshold =0.1 and length-two-loops-threshold =0.1 a process model (cf. Figure 3.17) with much higher number of connections than in the process model in Figure 3.16 was obtained.

The behaviour as seen in the process model of Figure 3.17 is relatively more detailed as compared to the process model mined with default parameter settings (Figure 3.16). The model is more informative in terms of the dependencies that it shows, but still some connections are missing. This problem has been observed in previous experiments too.

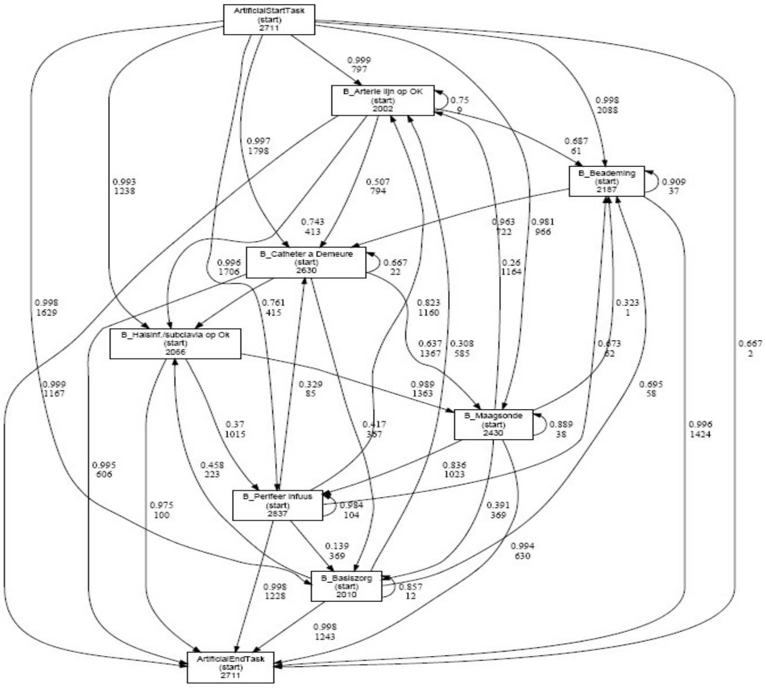


Figure 3.17: Treatments log mined with changed parameter settings

Moreover, from the split/join semantics for process model in Figure 3.16, it was also discovered that an activity does not represent clear AND/XOR join/splits. It was observed that the activities are not, only in AND or only in XOR relation with other activities i.e. AND and XOR may be mixed. For example, the outgoing connections of *ArtificialStartTask* involves a mix of AND/XOR activities and some tasks like *B_Arterie lijn op OK (start)* occur in more than one pair. Though this is a characteristic typical to the hospital domain (as the domain is quite flexible and hence the activities are not clearly in parallel or in choice with other activities) but it clearly depicts the limitation of the HM to show an activity both in AND, XOR relation at the same time. The presence of unclear AND/XOR joins/splits creates problems while parsing and therefore, the label on an arc in the dependency graph is lower than the frequency of the corresponding task (the activity *B_Cathether a Demeure* is executed 2630 times as indicated by the log but the label on the arc shows 1539 indicating that rest of its connections are missing in the process model). The fitness of the model is quite high: 0.78 indicating that 78% of the log traces are successfully parsed by the mined model, but in presence of problems like missing connections and unclear AND/XOR joins/splits, the fitness value seems to lose its importance as a measure depicting the quality of the mined model. In this example, the fitness of the model is quite high but the model itself is full of problems. In this situation the value of the fitness measure is questionable.

This experiment concludes the experimental illustrations. In the next subsection, the observations and findings from these experiments are summarized.

3.4.4 Summary: Experiments and Observations

The experiments conducted in the previous subsections were aimed at obtaining insights into the healthcare processes and evaluating the HM algorithm for these processes. We also understood the effect of parameter settings on the output of the algorithm. Below we summarize our observations and findings from these experiments:

1. The models for the healthcare processes contain complex spaghetti-like structures. The algorithm generates models for the input it is provided. In this case the inputs are the various healthcare logs. These logs illustrate the flexible and less-structured processes of the healthcare domain. Owing to these characteristics of the domain, the focus of this research assignment was to obtain simpler

models that could be analyzed for extracting meaningful information about the underlying processes concerning various patients. The HM however does not simplify its output for the complex input it is provided. We see the algorithm as an academic approach designed for research domain but it is not suitable for mining unstructured processes like healthcare. In unstructured processes the control flow paths do not have fixed form, and this flexibility leads to the complex process models.

2. Problems like: dangling activities, missing connections, missing activities are found in almost all the experiments done with the healthcare logs. Many of these problems can be attributed to the presence of noise or low-frequent behaviour. Low frequent behaviour in the healthcare domain represents exceptional medical cases. These exceptional medical cases can be of great interest but while capturing them the algorithm can also capture noise. It is unable to distinguish between noise and low frequent behaviour which puts a question mark on the behaviour captured in the mined model.
3. It was also seen that the algorithm generates different models for different parameter settings. Sometimes the settings produce the desired clean model but sometimes the existing activities and connections in the model are also lost. It is also observed that the number of parameters affecting the output of the algorithm is too large. This dependency of the algorithm on its various parameters leads to further problems like missing dependencies/activities, dangling activities and confusion whether which process model to trust as well as the large number of parameters makes it difficult and confusing for a user to obtain his desired model. Moreover, it is also not possible to reach to an optimum parameter setting for all the event logs.
4. It was also discovered that the activities in the healthcare log cannot be always characterized as clear AND/XOR join/splits. They sometimes belong to both of them. This typical characteristic of healthcare domain is not captured in the heuristics net provided by the algorithm.
5. It was also observed that when the parameter *all-activities-connected heuristic* is not used, the algorithm can generate better and simple models for some logs but this can not be generalised. For example, in case of complications log without a unique start and end point, the process model obtained was just a collection of unconnected activities. Whereas, for treatments log without start and end point, though the model consisted of dangling and unconnected activities it was simple to understand and conveyed some information about the underlying process. The structure of this model was not huge and confusing. But as already mentioned this parameter does not give desired and informative models for all logs. So, it can not be concluded whether not using the *all-activities connected heuristic* is a good choice.

Based on the above observations and findings, we conclude that the HM may not be the appropriate algorithm to gain insights into the processes of the healthcare domain. We are also convinced that the heuristics net representation used in the HM is not suitable for healthcare domain because it is unable to represent mixed AND and XOR situations discussed before. Therefore some alternate process model representations overcoming these limitations must be explored. In this context, we found the Disjunctive Workflow Schema (DWS) algorithm interesting because it not only gives visual process models but also rules which represent implicit behavioural pattern present in the log. This algorithm is implemented in ProM as DWS mining plug-in. We elaborate and investigate this algorithm in the next section.

3.5 The Disjunctive Workflow Schema algorithm

The DWS approach attempts to provide insights into a process whose enactment is constrained by some kind of rules, possibly involving information that is beyond the pure execution of activities [12]. It accepts an event log as an input and finds behavioural patterns in the log in form of rules known as discriminant rules. These rules are representations of constraints in the event log, which otherwise go unnoticed and undiscovered by other mining algorithms. This plug-in calculates these rules over projected traces in the log, and further they are used for partitioning the event log into variants. These rules are defined as follows:

"A discriminant rule is a rule of the form, [a1….ah]-/->a such that

- *The frequency of [a_1…..a_h] and [a_ha] in the log is over a given threshold sigma, i.e. they are both highly frequent and,*
- *The frequency [a_1….a_ha] is below a given threshold gamma, i.e. it is lowly frequent."*

For example, let us consider a discriminant rule: *a*, *b* -/-> *d* where *a*, *b* and *d* are activities in an event log. The rule states that the frequency of the activity sequence *ab*, and *bd* of the activities *a*, *b* and *b*, *d* respectively is above a threshold value (called *sigma*) specified by the user, but the frequency of the activity sequence *abd* of the tasks *a*, *b* and *d* together is less than a user specified threshold (called *gamma*).

We describe this approach in simple words as below:

1. Input: Event log of a process
2. First step is to discover the overall workflow schema[14]
3. Iteratively refine this schema by:
    a. Finding discriminant rules
    b. Cluster the traces characterized by these rules.
4. Use some mining algorithm to generate process models for these clusters.
5. The overall workflow schema then is the set of all the process model variants generated in step 4.

The DWS plug-in was designed to discover both, the control flow of a given process and the interesting global constraints which presents a refined view of the process. Traditionally the control flow perspective prescribes only the local constraints and misses out on the global ones. The local constraints are in form of relationships of precedence of tasks in a process, viz., an AND-join activity is executed only after all its predecessors are completed etc. Global constraints are richer in nature and their representation strongly depends on the particular application domain of the modelled process. The basic idea of the DWS approach is to first derive from the event log an initial workflow schema whose global constraints are left unexpressed and, then, to stepwise refine it into a number of specific schemas, each one modelling a class of trace having the same characteristics with respect to global constraints [12]. In the next section an example illustrates the working of the DWS algorithm.

3.5.1 The DWS plug-in

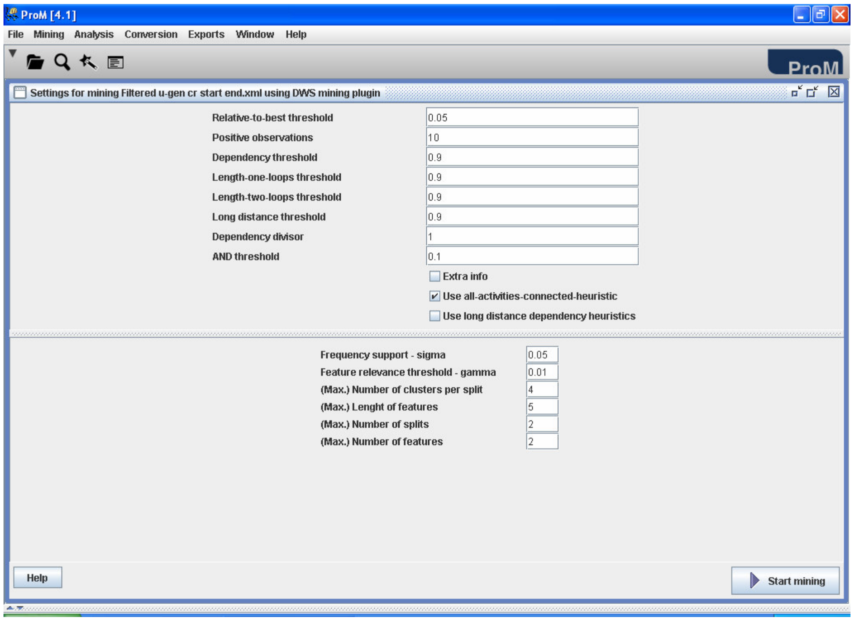
Figure 3.18 shows a screenshot of the DWS plug-in. Two frames divided by a separator can be seen in this figure. The parameters for the DWS plug-in are located in the bottom frame and in the top frame as we can recall are the parameters from the Heuristics Mining plug-in. The HM is used to construct the initial workflow schema and the process models from the traces characterized by different discriminant rules generated by the DWS algorithm. In spite of the limitations of the HM (cf. Section 3.4.4), the DWS plug-in based on the HM was chosen because the HM is robust to noise and imbalance. Besides, we wanted to study an algorithm that provides some alternate process model representations. The DWS besides providing the process model in form of the dependency graph also provides discriminant rules which convey behavioural information about the process.

In Figure 3.18 it can be seen that the values of the frequency thresholds: *gamma* and *sigma* can be specified by a user. Now as it is known that the DWS algorithm first derives an overall workflow schema of the underlying process, and then this schema is iteratively refined and clustered (using the *k*-means clustering algorithm [19]) on the basis of these rules, so the number of required refinements can also be specified by the parameter *Number of splits*. The parameter *Number of clusters per split* specifies the maximum number of *k* clusters to be used in the K-means algorithm. The number of rules to be mined as well as their length can be specified through the parameters: *Number of features* and *Length of features* respectively. The default values of all these parameters can be seen in the Figure 3.18. Figure 3.19 shows the results of mining the complications log (also used in Section 3.4.3, Illustration 3) using the DWS approach. The global workflow schema is represented by R. Two discriminant rules are discovered for this initial workflow schema and the process model characterizing these rules is also shown on the frame at the RHS. Let us understand one of these rules. Consider the following rule:

C_Colitis, pseudomembraneus, C_-VKF, atrium-flutter -/-> C_-SVT, paroxysmaal

[14] Workflow schema is the static aspect of a workflow process. It specifies which steps are required in the process, and in what order they should be executed. It is usually modelled as a directed graph defining this order of execution among the activities [35].

The rule can be interpreted as: ”the tasks (complications in this case) *C_Colitis, pseudomembraneus* and *C_-VKF, atrium-flutter* occur in this sequence more than 5% (the value of sigma is 0.05) in the event log as well as the tasks *C_-VKF, atrium-flutter* and *C_-SVT, paroxysmaal* also occur more than 5% in the log, but their combination in the same order occurs less than 1% in the event log. Although the mined process model allows for this behaviour, in context of the healthcare it can be said that in the event log the patients suffering from the complications: *C_Colitis, pseudomembraneus* and *C_-VKF, atrium-flutter*, and the patients suffering from the complications *C_-VKF, atrium-flutter* and *C_-SVT, paroxysmal* are highly frequent (found in 5% of the log traces) but the number of patients suffering from all three complications (in the same order) are low frequent (found only in 1% of the log traces).



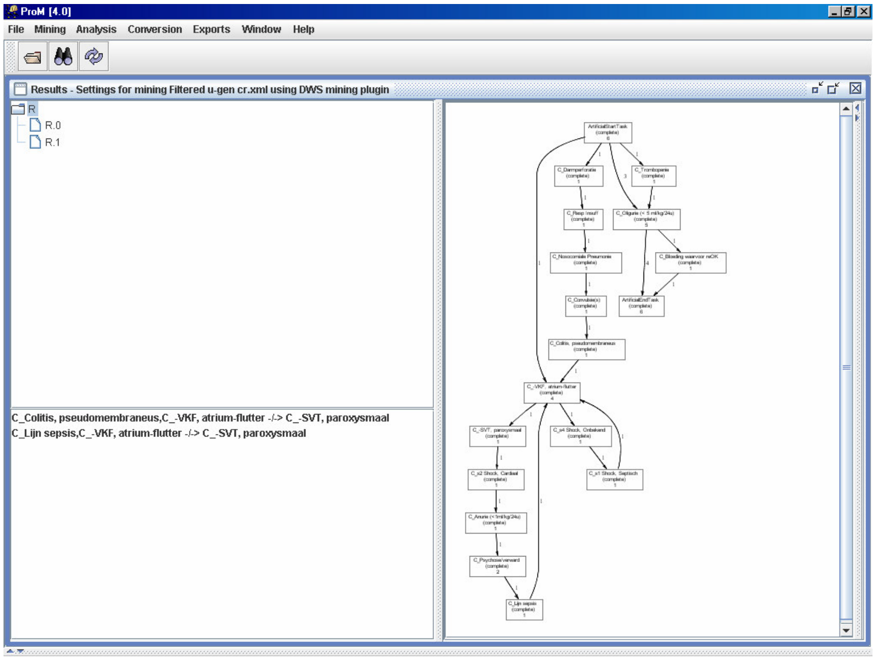
Settings for mining Filtered u-gen cr start end.xml using DWS mining plugin

Relative-to-best threshold	0.05
Positive observations	10
Dependency threshold	0.9
Length-one-loops threshold	0.9
Length-two-loops threshold	0.9
Long distance threshold	0.9
Dependency divisor	1
AND threshold	0.1

☐ Extra info
☑ Use all-activities-connected-heuristic
☐ Use long distance dependency heuristics

Frequency support - sigma	0.05
Feature relevance threshold - gamma	0.01
(Max.) Number of clusters per split	4
(Max.) Lenght of features	5
(Max.) Number of splits	2
(Max.) Number of features	2

Figure 3.18: Parameters for the DWS plug-in

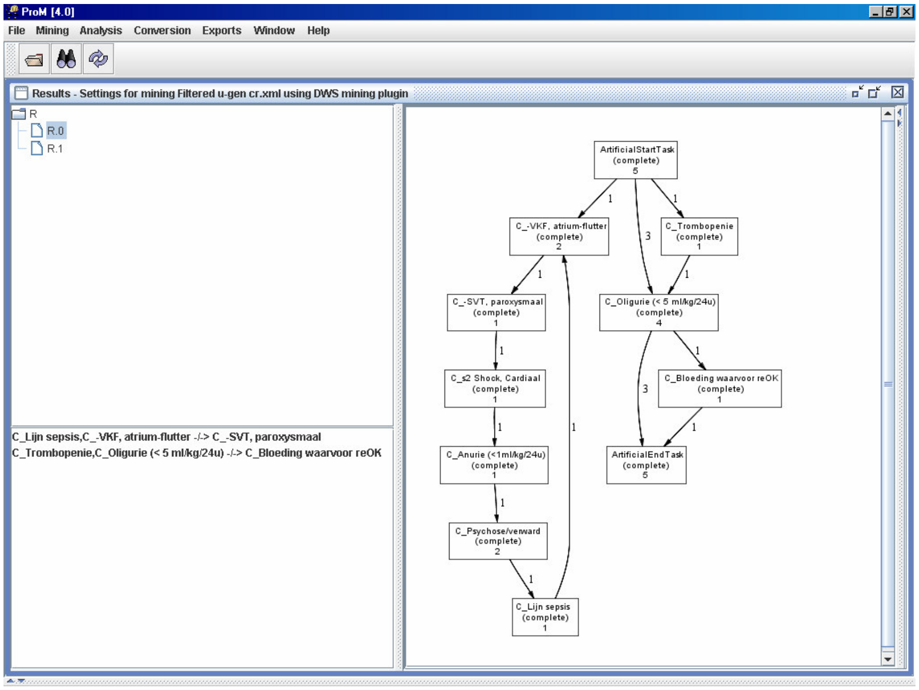

ProM [4.0]
File Mining Analysis Conversion Exports Window Help
R
R.0
R.1
ArtificialStartTask (complete) 6
C_Damperforatie (complete) 1
C_Trombopenie (complete) 1
C_Resp Insuff (complete) 1
C_Oligurie (< 5 ml/kg/24u) (complete) 5
C_Nosocomiale Pneumonie (complete) 1
C_Bloeding waarvoor reOK (complete) 1
C_Convulsie(s) (complete) 1
ArtificialEndTask (complete) 6
C_Colitis, pseudomembraneus (complete) 1
C_-VKF, atrium-flutter (complete) 4
C_-SVT, paroxysmaal (complete) 1
C_s4 Shock, Onbekend (complete) 1
C_s2 Shock, Cardiaal (complete) 1
C_s1 Shock, Septisch (complete) 1
C_Anurie (< 1ml/kg/24u) (complete) 1
C_Psychose/verward (complete) 2
C_Lijn sepsis (complete) 1
1
3
4

Results - Settings for mining Filtered u-gen cr.xml using DWS mining plugin

C_Colitis, pseudomembraneus,C_-VKF, atrium-flutter -/-> C_-SVT, paroxysmaal
C_Lijn sepsis,C_-VKF, atrium-flutter -/-> C_-SVT, paroxysmaal

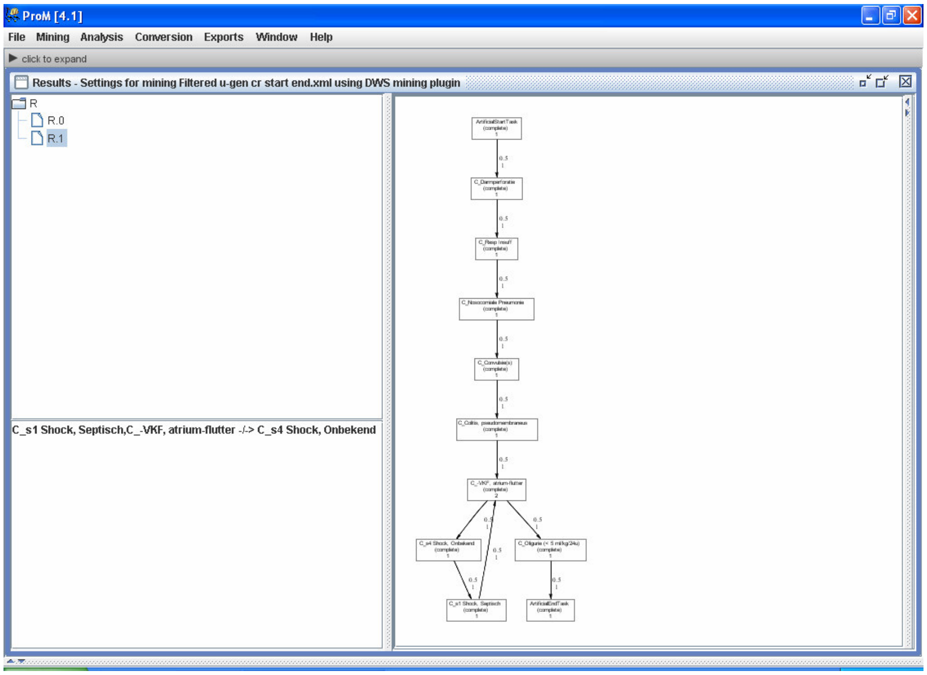
Figure 3.19: Discriminant rules and a process model as output of the DWS plug-in

This initial workflow schema R is further refined in two clusters: R.0 and R.1. Figure 3.20 and 3.21 shows the variants of the process model obtained from refining R and characterized by the discriminant rules.


ProM [4.0]
File
Mining
Analysis
Conversion
Exports
Window
Help
Results - Settings for mining Filtered u-gen cr.xml using DWS mining plugin
R
R.0
R.1
ArtificialStartTask (complete) 5
C_-VKF, atrium-flutter (complete) 2
C_Trombopenie (complete) 1
C_-SVT, paroxysmaal (complete) 1
C_Oligurie (< 5 ml/kg/24u) (complete) 4
C_s2 Shock, Cardiaal (complete) 1
C_Bloeding waarvoor reOK (complete) 1
C_Anurie (<1ml/kg/24u) (complete) 1
ArtificialEndTask (complete) 5
C_Psychose/verward (complete) 2
C_Lijn sepsis (complete) 1
1
1
3
1
1
1
1
3
1
1
1
1
1

C_Lijn sepsis,C_-VKF, atrium-flutter -/-> C_-SVT, paroxysmaal
C_Trombopenie,C_Oligurie (< 5 ml/kg/24u) -/-> C_Bloeding waarvoor reOK

Figure 3.20: The cluster R.0


ProM [4.1]
File
Mining
Analysis
Conversion
Exports
Window
Help
click to expand
Results - Settings for mining Filtered u-gen cr start end.xml using DWS mining plugin
R
R.0
R.1
ArtificialStartTask (complete) 1
C_Damperforatie (complete) 1
C_Resp Insuff (complete) 1
C_Nosocomiale Pneumonie (complete) 1
C_Convulsie(s) (complete) 1
C_Colitis, pseudomembraneus (complete) 1
C_-VKF, atrium-flutter (complete) 2
C_s4 Shock, Onbekend (complete) 1
C_Oligurie (< 5 ml/kg/24u) (complete) 1
C_s1 Shock, Septisch (complete) 1
ArtificialEndTask (complete) 1
0.5 1

C_s1 Shock, Septisch,C_-VKF, atrium-flutter -/-> C_s4 Shock, Onbekend

Figure 3.21: The cluster R.0

These discriminant rules express the behavioural patterns amongst the log activities. If algorithms like the α-algorithm or the HM were used on a log, only process models would have been generated. These process models depict the local constraints in form of control flow. The DWS approach also expresses the global constraints in form of discriminant rules. We found this as the strength of the DWS approach that it discovers both, the global as well as the local constraints from an event log. As opposed to the complex and huge process models generated by the Heuristics Mining plug-in, the DWS plug-in generates simpler and easy to comprehend process models for the sub-clusters (like R.0, R.1 etc.). But as the DWS uses the HM to generate the process models, it inherits the problems of the HM algorithm too. The user might also be prompted to experiment with different parameter settings of the HM, which can be quite tricky as the changes in the parameters may not always lead to desired results. However, the advantage of the DWS plug-in over the HM algorithm is that smaller process models are obtained based on some behavioural pattern specific to a particular process model (variants of the entire schema). And even if complex models are obtained for sub-clusters (in this case we used a small and filtered log therefore we obtained simple models at the sub-clusters), the rules generated by the plug-in provide some kind of behavioural information about the process. These behavioural patterns serve two purposes: first they are used as the basis of clustering the traces and second they represent some kind of information about the activities in the event log. Besides these advantages, the DWS has certain limitations, discussed in the next section.

3.5.2 Observations

This section mentions some of the limitations that were discovered in the DWS plug-in:

1. The discriminant rules are not simple to understand at a first look. For example, if we derive a rule like "*C_Lijn sepsis,C_-VKF, atrium-flutter -/-> C_-SVT, paroxysmal"* from the medical data and present it to the stakeholders they would find this rule difficult to comprehend as it deals with parameters associated with frequency. The stakeholders at large are not technical people, they can be doctors, and other staff from the hospital who would like to benefit from this rule. But to understand and use the knowledge represented by this rule, they have to understand the threshold values- sigma and gamma, otherwise it is difficult for them to understand the rule.
2. The discriminant rules deal only with neighbouring tasks. This is a shortcoming as it gives the relationship of a task only in context of its neighbour and then its relationship with non-neighbouring task is neglected. It seems like loss of information or lack of information as a task may also be related to other tasks that are not its neighbours. In context of healthcare, this incomplete information may be dangerous in place of being beneficial. If the relationship of some complication task is known only in context of complications that directly precede or follow it, and no information about its relationship with other complications is given, this information is not useful for the stakeholders from the medical domain.
3. Although the workflow schemas which are iteratively generated and guided by the notions of completeness and soundness[15], it is difficult to know the relative importance of different rules that are generated. Though these rules are ordered based on their importance but their importance is not quantified. The rules are not accompanied by some metric that depicts their importance in comparison with the other generated rules. It is only known that from top to bottom the importance decreases but how much is unknown. So though, the plug-in orders the rules, it lacks the quantification of their importance.
4. Currently the DWS mining plug-in uses the Heuristics Mining plug-in. Therefore, the problems faced with the HM in context of healthcare will also have to be dealt while using the DWS plug-in.

3.6 Conclusion

In this chapter we introduced the HM algorithm and illustrated our experiments with it in order to achieve one of the research goals for this thesis. Section 3.4.1 listed the limitations of this algorithm and this formed the motivation for investigating the DWS approach in Section 3.5. It was found that the strength of the DWS plug-in lies in its discovery of global and local constraints. The global constraints are discovered in form of discriminant rules and the local constraints are comprised in the variants of the process model represented by the various clusters. Limitations of this approach were stated in Section 3.5.2. Therefore it seems that we need to look at some other alternate process model representations which overcome the limitations of both the HM as well as the DWS algorithm. In this direction, we would explore the classical data mining concept of Association Rules in the next chapter, but we do this in the context of the ProM

[15] For the notions of soundness and completeness readers are referred to Appendix H.

framework. The Case Data Extraction (CDE) mining plug-in implemented in the framework can be used to experiment with the Association Rules in the Weka machine learning library. We elaborate on these concepts in the next chapter.

4 Mining Association Rules outside ProM

In the previous chapter, the Heuristics Mining and the DWS mining plug-ins were explained. The HM can be used to construct a process model reflecting the control flow behaviour that has been observed and recorded in an event log. The DWS algorithm generates process models for a cluster of PIs representing distinct discriminant rules. These discriminant rules identify structural patterns that are found in the process model but not registered in the log. The DWS uses the HM to construct process models. Experiments with these algorithms did not provide us with clean and desired process models which could be analyzed to obtain insight into the healthcare processes. The resulting process models were spaghetti-like structures. We concluded in Section 3.6 that these algorithms are not suitable for mining less structured processes of healthcare. Given the characteristics of the domain, it is imperative that any mining approach should focus on simplicity of results.

The limitations of the above mentioned plug-ins and the need for simple models for the healthcare domain led us to explore an approach that may give simple models as well throw light on behavioural patterns implicitly registered in the event log. Nowadays, the application of machine learning algorithms has become a widely adopted means to extract knowledge from vast amounts of data [22]. Combing through the machine learning algorithms, association rules were found to have potential to gain knowledge about the process and/or to make tacit knowledge explicit. They are simple to understand (as compared to the confusing and complex dependency graphs from the HM) and express behavioural (frequent) patterns in the log. This chapter therefore focuses on the third research goal of the thesis:

Investigate the usefulness of Association Rules as an alternate process model representation.

The chapter is organized as follows. First, Section 4.1 introduces the concept of association rules. Then, Section 4.2 explains the use of the Weka machine learning library to derive association rules. The experimental results for some healthcare logs are highlighted in Section 4.3. Section 4.4 concludes this chapter by summarizing the findings and observations of the experiments conducted with Weka.

4.1 Association Rules

Today Information and Communication Technologies are widely used in enterprises to maintain every record of their interactions with a client or prospect. These records can be seen as a learning opportunity [8] if this data gathering process leads to data analysis. This process of data analysis allows us to comb through the data for noticing patterns, devising rules, coming up with new ideas, figuring out the right questions, and making predictions about the future. For instance, the records can be analyzed for learning patterns for various transactions, viz., a pattern revealing a customer’s buying preferences, a reader’s visit to a website’s specific section etc. This analysis is the focus of data mining domain. Data mining, also known as Knowledge Discovery in Databases (KDD) is defined as [30]:

Data Mining is the process of discovering patterns in data.

There are various tasks that can be performed using data mining techniques, viz., classification, estimation, prediction, affinity grouping, clustering, and description and profiling tasks. But our focus is on affinity grouping or association rules. The task of affinity grouping or association rules is to determine which things go together [8]. Association rules tell us about the association between two or more items/elements/tasks in a database. The Market Basket Analysis (MBA) is the largest application for algorithms discovering these association rules. It is a modelling technique based upon the theory that if a person buys a certain group of items, he/she is more (or less) likely to buy another group of items [31]. The MBA is based on discovering purchasing habits of the customers and the association between different items that customers place in their “shopping baskets”. A sample association rule is given below:

Bread, Milk=> Butter | 90%

The items on LHS {Bread, Milk} of an association rule are called antecedents and the items {Butter} on the RHS are called consequents. An association rule can have multiple antecedents and multiple consequents. The 90% factor in the above rule indicates that 90% of the customers who bought bread and milk also bought butter. This percentage indicates the *certainty* or the *confidence* of this association rule. The confidence factor is one of the measures of the interestingness of an association rule. Another measure is the *support*. Support indicates the usefulness of an association rule. For example, if the above rule has a support of 5% it means that 5% of all the transactions under analysis show that bread, milk and butter are purchased together. When the technique of association rules is applied to event logs, we would like to retrieve associations and frequent patterns existing amongst the various events in event logs. This is described in detail in the coming sections, but before that some important concepts related to the association rules viz., support, confidence etc. are explained.

4.1.1 Definitions: Association rules, Support & Confidence

1. Association rules:

Association rules are formally defined as statements of the form X=> Y where X and Y are disjoint itemsets i.e., $X \cap Y = \phi$, and Y is a non empty itemset. X and Y are sets of items from the transactional data. This rule holds in a transaction set D with confidence c if c% of transactions in D that contain X also contain Y. The rule X=>Y has support s in the transaction set D if s% of transactions in D contains X U Y. Association rules suggests a strong co-occurrence relationship between items in antecedent and consequent of the rule. They do not necessarily imply causality.

2. Support

In many situations, association rules involving sets of items that appear frequently in a database or transaction log are of interest. This means that only the items with high support are interesting. In absolute terms, support of an item is the number of times this item appears in a log. For the association rule: a=>b, support can be understood as the joint probability of a and b. It indicates the coverage of a rule i.e. how often a rule is applicable to a given dataset. An itemset (set of items) satisfying a minimum support value is referred to as frequent itemset or large itemset. This minimum support value is called the minimum support threshold. Support can be calculated as:

$$Support(a,b) = \left(\frac{\#tuples(a,b)}{\#tuples} \right)$$

Equation 4.1: Support of an itemset

3. Confidence

Confidence of an association rule X=>Y is the probability of finding Y in the transaction set D. In simple words, it indicates how frequently items in Y appear in transactions that contain X. It is also referred to as the accuracy of the association rule. For example, for an association rule involving items a and b: a =>b, the confidence is the conditional probability of 'b' given 'a', i.e. how much percentage of transactions in the database that has item 'a' also contains item 'b'. It is given as:

$$Confidence(a,b) = \left(\frac{\#tuples(a,b)}{\#tuples(a)} \right)$$

Equation 4.2: Confidence of an association rule

Confidence can also be derived from Equation 4.1:

$$Confidence(a,b) = \frac{Support(a \bigcup b)}{Support(a)}$$

Equation 4.3: Confidence of a rule can be derived using support count

Support indicates how useful is the rule and confidence indicates how strong is the rule or how certain we are of it. Below we give an example dataset and illustrate the calculation of support of an item and confidence of a rule in this dataset.

Table 4.1: Example dataset to illustrate support and confidence measures

Transaction ID	List of Items
101	pen, paper
102	pen, pencil, eraser
103	pencil, drawing sheets
104	pencil, eraser
105	pencil, notebook, eraser
106	paper, pencil
107	pen, paper, calculator
108	pencil, paper, calculator
109	drawing sheets, pencil, eraser
110	paper, pencil, eraser

In the above table, the item {pencil} has a support of 8 as it appears in transactions: 102, 103, 104, 105, 106, 108, 109, 110. The itemset {pencil, eraser} has a support of 5 since it appears in transactions: 102, 104, 105, 109 and 110. When we use the support count we are refer to absolute support. The corresponding relative support of item {pencil} is 8/10 = 80% as out of total 10 transactions the item {pencil} appears in 8 transactions. The confidence for the rule {pencil} => {eraser} is 62.5% because 5 of the transactions that includes {pencil} also includes {eraser}.

4. Predictive Accuracy

Different association rule algorithms use different metric to determine interestingness of any association pattern. The Apriori algorithm uses the metrics: Support and Confidence for this purpose. Another measure is Predictive Accuracy. It is an indicator of a rule's accuracy in future over unseen data. Confidence of a rule is the ratio of the correct predictions over all records for which a prediction is made but it is measured with respect to the database that is used for training. This confidence on the training data is only an estimate of the rule's accuracy in the future, and since we search the space of association rules to maximize the confidence, the estimate is optimistically biased [24]. Thus, the measure predictive accuracy is introduced. It gives for an association rule its probability of a correct prediction with respect to the process underlying the database.

4.1.2 Algorithms for Association Rules

This section explains methods that generate association rules. All algorithms for association rule mining involves two steps:

1. *Find all frequent itemsets*. According to the definition in Section 4.1.1, these itemsets will occur at least as frequently as a predetermined minimum support count.
2. *Generate strong association rules from the frequent itemsets*. In this step, strong association rules are derived from the frequent itemsets generated in the first step. Strong rules are the rules that satisfy both a minimum support threshold and a minimum confidence threshold. They are preferred because it is not practical to do an exhaustive search for thousands of potential rules that can be generated from a database. Many of these rules will not be of interest and use because they may be unreliable due to low support or confidence values. Therefore it is common to generate only those rules that have a minimum specified support and confidence values.

Association rules can be discovered using algorithms like the Apriori, AprioriTid, PredictiveApriori, Tertius etc. Below we give a brief description of some of these algorithms[16].

1. The Apriori algorithm

The Apriori algorithm was proposed by R. Agrawal and R. Srikant [7] in 1994 for mining frequent itemsets for Boolean association rules. Boolean association rules are the rules involving associations between the

[16] The pseudo code for the algorithms explained in this section can be found in Appendix I.

presence/absence of items. A rule describing associations between quantitative items or attributes is called a quantitative association rule.

The Apriori algorithm uses a level-wise search to generate frequent itemsets traversing from frequent 1-itemsets (an itemset containing k items is referred to as a k-itemset) to the maximum size of frequent itemsets. This iterative search is continued till no new frequent itemsets can be generated. The steps of the Apriori algorithm are given below:

1. For the given support threshold s, in the first pass find the items with pre-specified minimum support value. The resulting set is denoted by L_1.
2. Pairs of L_1 are the candidates for itemsets of size 2. These candidates are denoted by C_2. The frequent itemset pairs that satisfy the support count s are the frequent itemsets of size 2 and are contained in L_2.
3. The pairs in L_2 are the candidates for frequent itemsets of size 3. From L_2 size 3-itemsets are formed by comparing itemsets differing only in the last item and are in lexicographic order. Again the size3-itemsets satisfying the support count are contained in C_3.
4. From C_3 we again construct L_4 and we can keep on proceeding like this till no further frequent itemsets can be generated based on the support count measure. But we know L_i is the set of frequent itemsets of size i, C_{i+1} is the set of candidate frequent itemsets of size i+1 such that each subset of size i is in $L_{i.}$
5. Once these frequent itemsets are obtained, for each of the frequent itemset l, generate all nonempty subsets of l.
6. For every nonempty subset s of l, we get the rule "s=> (l-s)" if the ratio of support (l) to support (s) is greater than or equal to minimum confidence threshold value.

2. The AprioriTid algorithm

Both the Apriori and the AprioriTid [7] algorithms generate the candidate itemsets to be counted in a pass by using only the itemsets found large in the previous pass-without considering the transactions in the database. The AprioriTid algorithm has an additional property that the database is not used at all for counting the support of candidate itemsets after the first pass. Rather than using the database transactions, this algorithm uses the entries in $\overline{c_k}$ to count the support of candidates in C_k. $\overline{c_k}$ is the set of candidate k-itemsets when the transaction IDs of the generating transactions are kept associated with the candidates. Keeping a track of transactions IDs from which the candidate frequent itemsets are generated at each level greatly reduces the reading effort in later passes. Once these candidate itemsets are obtained, association rules can be found just like in the Apriori algorithm (Steps 5 and 6).

3. The PredictiveApriori algorithm

PredictiveApriori algorithm combines the measures of support and confidence into a single measure referred to as the predictive accuracy and finds the best n association rules in order. Confidence of a rule can be understood as the ratio of correct predictions over all records (transactions) for which a prediction is made. The PredictiveApriori algorithm computes the values of support and confidence without taking these values from the user and the value of predictive accuracy is found from their values. The predictive accuracy c for association rule: a=>b is defined as the probability of a correct prediction with respect to the process underlying the database [24]. The algorithm uses the same logic to find the frequent itemsets and once the frequent itemsets are obtained, the PredictiveApriori algorithm finds the n best rules based on the values of predictive accuracy.

The Weka machine learning library provides various association rule algorithms. It is a collection of machine learning techniques[17] and data pre-processing tools. Readers are referred to Appendix J for an

[17] Machine learning is the study of computer algorithms that improve automatically through experience. Applications of machine learning range from data mining programs that discover general rules in large data sets, to information filtering systems that automatically learn users' interests [20]. Readers are referred to [20, 30] for further reading about machine learning.

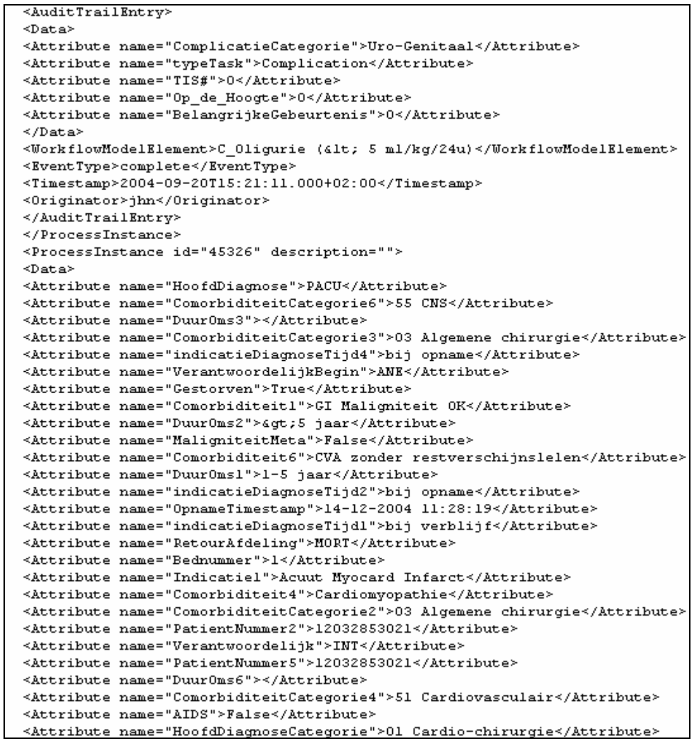
overview of the Weka library. In the next section, Weka is introduced and it is discussed how it can be used to obtain association rules for healthcare processes.

4.2 Association rules and Weka library

4.2.1 Introduction

As already mentioned in Section 4.1, data mining is the exploration and analysis of large quantities of data in order to discover meaningful patterns and rules. It includes tasks such as classification, estimation, prediction, affinity grouping, clustering, and description and profiling tasks. The Weka workbench provides tools for performing all these tasks. It includes methods for all the standard data mining problems: regression, classification, clustering, association rule mining, and attribute selection. Our focus is on the generation of Boolean association rules for an event log.

To obtain association rules for tasks in an event log, this log is provided as input to Weka. The output of the Case Data Extraction (CDE) plug-in implemented in ProM is used as an input to Weka. The CDE (cf. Appendix K) converts the case data of an event log into a table. Case data refers to: *PI data attributes, ATE data attributes, originators and event types.* An excerpt of a healthcare log and its CDE table obtained from the CDE plug-in can be found in figures 4.1a) and 4.1b) respectively. When this table is exported to a *comma separated values format*[18] *(CSV)* file it can serve as an input to Weka[19]. The CDE table can be exported to a CSV file using the Standard CSV Export plug-in available in ProM. Figure 4.2 shows the main screen of Weka with this corresponding CSV file as input.

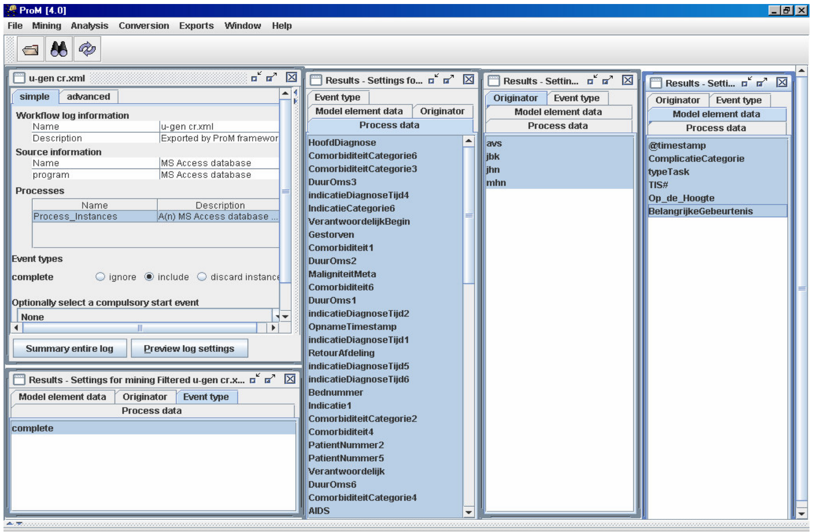


```
<AuditTrailEntry>
<Data>
<Attribute name="ComplicatieCategorie">Uro-Genitaal</Attribute>
<Attribute name="typeTask">Complication</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="Op_de_Hoogte">0</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>C_Oligurie (&lt; 5 ml/kg/24u)</WorkflowModelElement>
<EventType>complete</EventType>
<Timestamp>2004-09-20T15:21:11.000+02:00</Timestamp>
<Originator>jhn</Originator>
</AuditTrailEntry>
</ProcessInstance>
<ProcessInstance id="45326" description="">
<Data>
<Attribute name="HoofdDiagnose">PACU</Attribute>
<Attribute name="ComorbiditeitCategorie6">55 CNS</Attribute>
<Attribute name="DuurOms3"></Attribute>
<Attribute name="ComorbiditeitCategorie3">03 Algemene chirurgie</Attribute>
<Attribute name="indicatieDiagnoseTijd4">bij opname</Attribute>
<Attribute name="VerantwoordelijkBegin">ANE</Attribute>
<Attribute name="Gestorven">True</Attribute>
<Attribute name="Comorbiditeit1">GI Maligniteit OK</Attribute>
<Attribute name="DuurOms2">&gt;5 jaar</Attribute>
<Attribute name="MaligniteitMeta">False</Attribute>
<Attribute name="Comorbiditeit6">CVA zonder restverschijnslelen</Attribute>
<Attribute name="DuurOms1">1-5 jaar</Attribute>
<Attribute name="indicatieDiagnoseTijd2">bij opname</Attribute>
<Attribute name="OpnameTimestamp">14-12-2004 11:28:19</Attribute>
<Attribute name="indicatieDiagnoseTijd1">bij verblijf</Attribute>
<Attribute name="RetourAfdeling">MORT</Attribute>
<Attribute name="Bednummer">1</Attribute>
<Attribute name="Indicatie1">Acuut Myocard Infarct</Attribute>
<Attribute name="Comorbiditeit4">Cardiomyopathie</Attribute>
<Attribute name="ComorbiditeitCategorie2">03 Algemene chirurgie</Attribute>
<Attribute name="PatientNummer2">12032853021</Attribute>
<Attribute name="Verantwoordelijk">INT</Attribute>
<Attribute name="PatientNummer5">12032853021</Attribute>
<Attribute name="DuurOms6"></Attribute>
<Attribute name="ComorbiditeitCategorie4">51 Cardiovasculair</Attribute>
<Attribute name="AIDS">False</Attribute>
<Attribute name="HoofdDiagnoseCategorie">01 Cardio-chirurgie</Attribute>
```

Figure 4.1a: Case properties: PI and ATE data properties, originator and timestamp information can be seen in a fragment of a healthcare log

[18] A CSV file is a specially formatted plain text file which stores spreadsheet or basic database-style information in a very simple format, with one record on each line, and each field within that record separated by a comma.

[19] The CSV file from ProM cannot be directly used in Weka as it contains non-binary attributes too whereas Weka generates Boolean association rules that take up only binary information. So, this CSV file has to be modified. Readers are referred to Appendix L for these modifications.



Workflow log information

Name	u-gen cr.xml
Description	Exported by ProM framewor

Source information

Name	MS Access database
program	MS Access database

Processes

Name	Description
Process_Instances	A(n) MS Access database ...

Event types

complete ○ ignore ● include ○ discard instance

Optionally select a compulsory start event

None

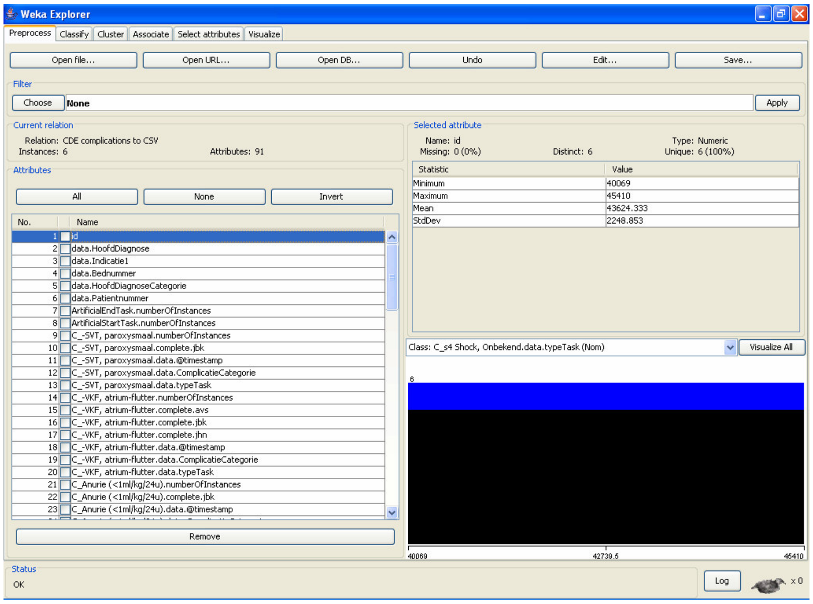
complete

HoofdDiagnose
ComorbiditeitCategorie6
ComorbiditeitCategorie3
DuurOms3
indicatieDiagnoseTijd4
IndicatieCategorie6
VerantwoordelijkBegin
Gestorven
Comorbiditeit1
DuurOms2
MaligniteitMeta
Comorbiditeit6
DuurOms1
indicatieDiagnoseTijd2
OpnameTimestamp
indicatieDiagnoseTijd1
RetourAfdeling
indicatieDiagnoseTijd5
indicatieDiagnoseTijd6
Bednummer
Indicatie1
ComorbiditeitCategorie2
Comorbiditeit4
PatientNummer2
PatientNummer5
Verantwoordelijk
DuurOms6
ComorbiditeitCategorie4
AIDS

avs
jbk
jhn
mhn

@timestamp
ComplicatieCategorie
typeTask
TIS#
Op_de_Hoogte
BelangrijkeGebeurtenis

Figure 4.1b: Case properties mapped onto a table can be obtained using the CDE plug-in



None

Relation: CDE complications to CSV
Instances: 6 Attributes: 91

No.		Name
1	□	id
2	□	data.HoofdDiagnose
3	□	data.Indicatie1
4	□	data.Bednummer
5	□	data.HoofdDiagnoseCategorie
6	□	data.Patientnummer
7	□	ArtificialEndTask.numberOfInstances
8	□	ArtificialStartTask.numberOfInstances
9	□	C_-SVT, paroxysmaal.numberOfInstances
10	□	C_-SVT, paroxysmaal.complete.jbk
11	□	C_-SVT, paroxysmaal.data.@timestamp
12	□	C_-SVT, paroxysmaal.data.ComplicatieCategorie
13	□	C_-SVT, paroxysmaal.data.typeTask
14	□	C_-VKF, atrium-flutter.numberOfInstances
15	□	C_-VKF, atrium-flutter.complete.avs
16	□	C_-VKF, atrium-flutter.complete.jbk
17	□	C_-VKF, atrium-flutter.complete.jhn
18	□	C_-VKF, atrium-flutter.data.@timestamp
19	□	C_-VKF, atrium-flutter.data.ComplicatieCategorie
20	□	C_-VKF, atrium-flutter.data.typeTask
21	□	C_Anurie (<1ml/kg/24u).numberOfInstances
22	□	C_Anurie (<1ml/kg/24u).complete.jbk
23	□	C_Anurie (<1ml/kg/24u).data.@timestamp

Name: id Type: Numeric
Missing: 0 (0%) Distinct: 6 Unique: 6 (100%)

Statistic	Value
Minimum	40069
Maximum	45410
Mean	43624.333
StdDev	2248.853

Class: C_s4 Shock, Onbekend.data.typeTask (Nom)

OK

Figure 4.2: The Weka explorer and the healthcare log shown in Figure 4.1a

Figure 4.2 shows the case properties, which were selected during the CDE mining. For example, *data.HoofdDiagnose, data.Indicatie1,* etc. are the data attributes of PIs in the log. Attributes like *numberOfInstances, timestamp, ComplicatieCategorie* etc. are the case properties of each ATE in the log. These attributes form the input for association analysis algorithms in Weka. In the next section, we show some association rules generated from the logs for Case study1 and evaluate if they can be useful in understanding healthcare processes.

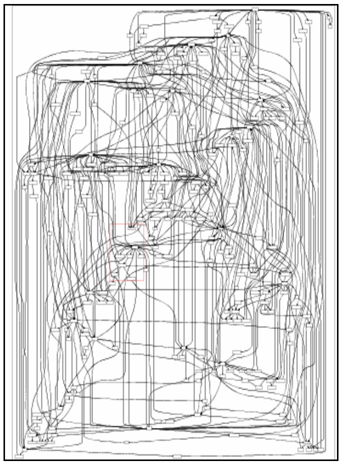
4.3 Experimental results

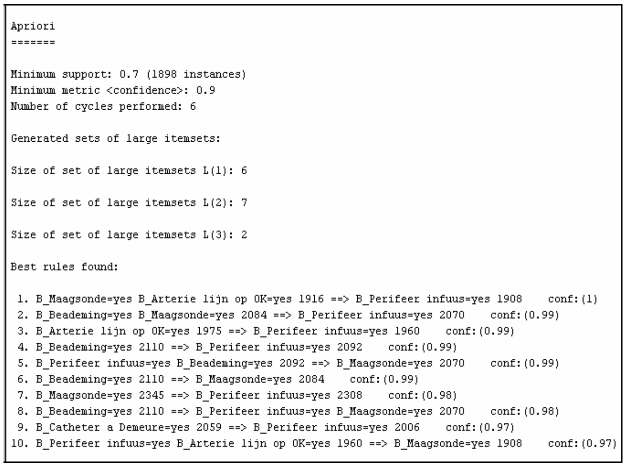
The CSV files obtained from ProM can be used for discovering association rules in the Weka library. The library provides the Apriori, PredictiveApriori and Tertius algorithms for association analysis. The purpose of this section is to analyze the usefulness of association rules in gaining insights into the less structured processes of healthcare. These rules would be evaluated on criteria like simplicity, ease of understanding and insights provided for the underlying process. It should be remembered that the motivation to explore these rules is to see how they fare on less structured processes in comparison to the mining plug-ins: HM and the DWS.

This section illustrates some experiments with the Case study1 and compares the results of these experiments for three mining approaches: The HM, the DWS and the association analysis. This will give an indication of the potential of these rules to extract behavioural knowledge from the healthcare event logs. These experiments are illustrated below:

4.3.1. Illustration 1

The healthcare log used in this experiment is about the treatments (cf. Section 3.4.1) that the patients receive. The log has 2711 PIs and 255 ATEs. We used this log for experimenting with the HM algorithm and the Apriori algorithm provided in Weka. Parameter settings used for the Apriori algorithm are: confidence= 0.9, support=0.7 and the desired number of rules =10. The HM was experimented with default settings. Following figure gives the process models generated by these two algorithms:





```
Apriori
=======

Minimum support: 0.7 (1898 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 6

Generated sets of large itemsets:

Size of set of large itemsets L(1): 6

Size of set of large itemsets L(2): 7

Size of set of large itemsets L(3): 2

Best rules found:

 1. B_Maagsonde=yes B_Arterie lijn op OK=yes 1916 ==> B_Perifeer infuus=yes 1908    conf:(1)
 2. B_Beademing=yes B_Maagsonde=yes 2084 ==> B_Perifeer infuus=yes 2070    conf:(0.99)
 3. B_Arterie lijn op OK=yes 1975 ==> B_Perifeer infuus=yes 1960    conf:(0.99)
 4. B_Beademing=yes 2110 ==> B_Perifeer infuus=yes 2092    conf:(0.99)
 5. B_Perifeer infuus=yes B_Beademing=yes 2092 ==> B_Maagsonde=yes 2070    conf:(0.99)
 6. B_Beademing=yes 2110 ==> B_Maagsonde=yes 2084    conf:(0.99)
 7. B_Maagsonde=yes 2345 ==> B_Perifeer infuus=yes 2308    conf:(0.98)
 8. B_Beademing=yes 2110 ==> B_Perifeer infuus=yes B_Maagsonde=yes 2070    conf:(0.98)
 9. B_Catheter a Demeure=yes 2059 ==> B_Perifeer infuus=yes 2006    conf:(0.97)
10. B_Perifeer infuus=yes B_Arterie lijn op OK=yes 1960 ==> B_Maagsonde=yes 1908    conf:(0.97)
```

Figure 4.3: The spaghetti-like process model from the HeuristicsMiner and the association rules from the Apriori

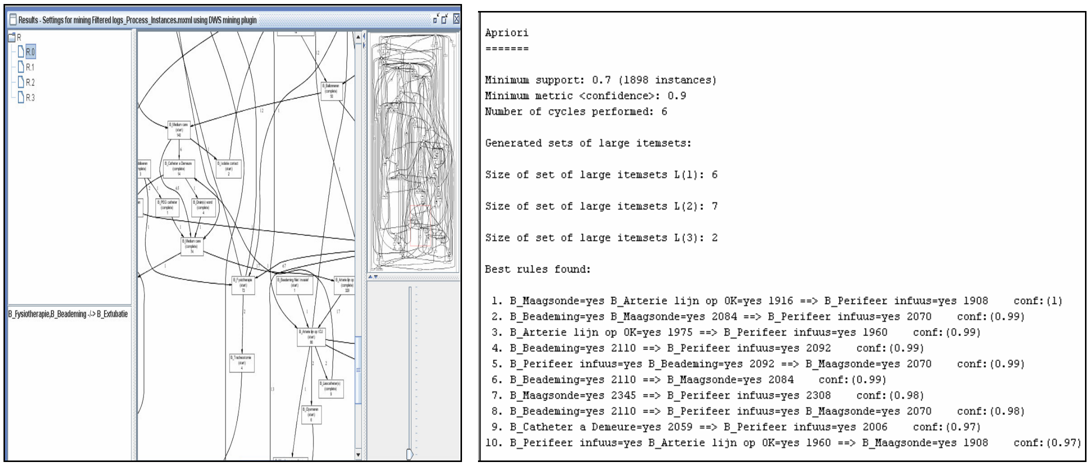
As apparent, the process model generated by the HM is complex and confusing. Undoubtedly the model represents the chaos, unstructuredness and flexibility in the treatment process. From this process model it is difficult to trace out even the treatment route followed by a single patient. The complexity and hugeness of this model restricts the stakeholders from extracting any information about the underlying process except the fact that the process has very little structure. Moreover, the process model is constructed from the causal dependencies between the activities in the log and it does not provide any information about the behavioural patterns existing implicitly in the event log.

In contrast to this model, the Apriori association analysis algorithm presents the process model in form of association rules. These rules as already mentioned are the behavioural patterns existing in the log but registered implicitly. If we analyze a rule, for example, ***B_Beademing=yes B_Maagsonde=yes 2084 ==> B_Perifeer infuus=yes 2070 conf :(0.99),*** it says, the treatments *B_Beademing* and *B_Maagsonde* are always eventually followed by the treatment *B_Perifeer infuus*. In the rule, the number before the arrow is the number of instances for which the antecedent is true and the number after the arrow is the number of instance in which the consequent is true; and the confidence (cf. conf) is the ratio between the two [28]. The Confidence of this rule is 0.99 indicating that in 99% of the log traces where the activities *B_Beademing* and *B_Maagsonde* occur, the activity *B_Perifeer infuus* also occurs. The stakeholders from the healthcare domain can benefit from such information because they can be well prepared to treat patients undergoing the treatments *B_Beademing* and *B_Maagsonde* (as they know the treatment *B_Perifeer infuus* will also be prescribed to these patients).

From this example it becomes apparent that the stakeholders (medical staff or researchers) can benefit better from association rules than the complex models as shown in Figure 4.3 as the rules are simple to understand and show a pattern present in the log. The rules as seen in the figure above are ranked according to their confidence. This gives an idea of the strength of different rules generated by the algorithm. Further, the rules do not exhibit problems like the unclear AND/XOR, dangling activities, missing dependencies which degrade the quality of the process model. This comparison between the HM and the association analysis technique leads to the fact that association analysis looks a promising approach for mining less structured processes. In the next section, we compare the DWS with the association analysis algorithms and evaluate if the association rules fare better than the discriminant rules.

4.3.2. Illustration 2

For this experiment the same log as used for Illustration 1 is used. The DWS algorithm and the Apriori algorithm are used on this log. Following figure gives the process models generated by these two algorithms:



Results - Settings for mining Filtered logs_Process_Instances.mxml using DWS mining plugin

B_Fysiotherapie,B_Beademing -/-> B_Extubatie

```
Apriori
=======

Minimum support: 0.7 (1898 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 6

Generated sets of large itemsets:

Size of set of large itemsets L(1): 6

Size of set of large itemsets L(2): 7

Size of set of large itemsets L(3): 2

Best rules found:

 1. B_Maagsonde=yes B_Arterie lijn op OK=yes 1916 ==> B_Perifeer infuus=yes 1908    conf:(1)
 2. B_Beademing=yes B_Maagsonde=yes 2084 ==> B_Perifeer infuus=yes 2070    conf:(0.99)
 3. B_Arterie lijn op OK=yes 1975 ==> B_Perifeer infuus=yes 1960    conf:(0.99)
 4. B_Beademing=yes 2110 ==> B_Perifeer infuus=yes 2092    conf:(0.99)
 5. B_Perifeer infuus=yes B_Beademing=yes 2092 ==> B_Maagsonde=yes 2070    conf:(0.99)
 6. B_Beademing=yes 2110 ==> B_Maagsonde=yes 2084    conf:(0.99)
 7. B_Maagsonde=yes 2345 ==> B_Perifeer infuus=yes 2308    conf:(0.98)
 8. B_Beademing=yes 2110 ==> B_Perifeer infuus=yes B_Maagsonde=yes 2070    conf:(0.98)
 9. B_Catheter a Demeure=yes 2059 ==> B_Perifeer infuus=yes 2006    conf:(0.97)
10. B_Perifeer infuus=yes B_Arterie lijn op OK=yes 1960 ==> B_Maagsonde=yes 1908    conf:(0.97)
```

Figure 4.4: A discriminant rule in the DWS and the association rules from the Apriori

The discriminant rule in the process variant R1 (cf. above figure) is:

B_Fysiotherapie, B_Beademing-/->B_Extubatie

It is interpreted as: The treatments *B_Fysiptherapie and B_Beademing;* and also the treatments *B_Beademing and B_Extubatie* are given to 50% of the total patients whose information is recorded in the event log, but these three treatments together are not given even to 10% of the total patients. These percentages represent the parameters: sigma and gamma of the DWS approach. The above interpretation of this discriminant rule is not straightforward to understand as it involves the understanding of these frequency related parameters. Moreover, the emphasis of these rules is also on the execution order of the activities which is also seen as a limitation of the rule in case the 'timestamp' information is missing from the log and the ordering of activities is based on only on the date of execution. This can create problems when multiple activities are registered on same date but timestamp information is not present. In this case, these rules will show incorrect behavioural pattern as they depend on the ordering of activities. It is also found that the rules do not take into account the event type information associated with the activities. In presence of multiple event types, it is not known which events (activity + event type) are included in the rule. This may hinder the complete understanding of the situation because it may be possible, for example that the start of one treatment and the end of another treatment are both frequent (it means that as many times one treatment finishes the same number of times the other treatment starts).

In contrast to the discriminant rules, the Apriori association rules are easy to understand. The relative importance of multiple discriminant rules for a process variant is not known. The confidence and predictive accuracy values are used for ranking the association rules in the Apriori and the PredictiveApriori algorithms. The DWS mining algorithm uses the HM algorithm for process discovery and as already said the HM is not an appropriate mining technique for healthcare processes. Also, the DWS analysis plug-in assumes that the mined process model is a dependency graph. This restriction makes the DWS plug-ins less flexible. The association rules however do not provide any visual process discovery, but they can be used as the basis for clustering. Later, any mining and analysis technique can be applied to these clusters. Moreover, the advantage of association rules is also reflected by the fact that they represent frequent patterns in the log. If the log contains complications, the association rules will depict the frequently occurring complications and this fact is strengthened by the statistical support provided by the confidence/predictive accuracy values. These rules in presence of adequate domain knowledge would definitely provide meaningful information about the underlying healthcare process thereby helping in improvements in the medical services.

4.4 Conclusion

The experimental results in the previous section favours the use of association rules for mining less structured processes like the healthcare processes. It should be noted that the above illustrations exhibit the use of data from Case study1. As indicated in Section 4.3, these rules would be evaluated on criteria of simplicity, ease of understanding and insights provided for the investigated process. It is clear from the above illustrations that the association rules are simple and easy to interpret, and they have the potential to provide knowledge in form of behavioural/frequent patterns existing implicitly in the log. This information can be of importance to the healthcare personnel who can make anticipated preparations in terms of skills and equipments needed to deal with any emergency or similar situation for example, where A, B and C’s absence implies execution of task C. The behavioural patterns (rules) combined with the domain knowledge would be of great use to interpret and understand the various activities happening in any healthcare organization

The simplicity of these rules and the knowledge derived by them motivated us to further explore the domain of association rules and to get insights into how they can be of use in mining less structured processes. Currently the output of the CDE can be used for generating association rules from Weka but this process is not straightforward and consumes time as the CDE output has to be modified before it can be used in the Weka library for association rule generation. Therefore, we propose to implement a new mining plug-in in the ProM framework that will generate association rules from an event log. We also propose to provide a mechanism that can make use of these rules to provide specific process models. The next chapter discusses this proposal and later, showcases the new mining plug-in.

5 Mining Association Rules inside ProM

As indicated in the previous chapter, this chapter discusses implementation of the new mining plug-in for generation of association rules within the ProM framework. This new plug-in is called the Association Rule Miner (ARM). We begin the chapter by motivating our choice for the algorithms chosen for implementation, followed by presenting the features of the new plug-in. In Section 5.2, the newly developed plug-in and experiments with it are presented. Performance of these algorithms on healthcare logs is analyzed in Section 5.3 and Section 5.4 concludes the chapter.

5.1 The Association Rule Miner (ARM) plug-in

Besides the algorithms discussed in Section 4.1.2, there are many other algorithms developed over time to generate association rules from transactional data. These algorithms can be found in [7, 9, 21]. For the new plug-in, the algorithms: the Apriori algorithm and the PredictiveApriori algorithm were chosen. The Apriori algorithm being the most basic algorithm giving insights into the method of finding frequent itemsets and association rules was chosen. The PredictiveApriori algorithm was chosen because there can be thousands of rules which can be generated from some transactional data but not all of them might be useful and interesting. So, there must be some mechanism to select some of the best rules. This is exactly what the PredictiveApriori algorithms offer. Based on the concept of predictive accuracy, it offers to the user *n* best rules where *n* is the number of desired rules and these rules are ranked in their order of importance based on their predictive accuracy.

5.1.1 The features of the ARM

In this section the salient features of the proposed ARM plug-in are presented. These are given below:

- The ARM mining plug-in aims at generating Boolean Association Rules from the event log supplied to the ProM framework as input. The actual association analysis in the ARM is provided by the Weka machine learning software library. As we know that these algorithms in Weka work on ARFF or CSV file format, so we need the conversion of an MXML input log to the Weka's input file format (ARFF/CSV). So, the ARM first converts the input MXML event log into ARFF learning instances.

- The ARM offers the full range of parameters that are available for the association rule algorithms from the Weka library. For example, for the Apriori algorithm it offers the parameters like confidence, support etc. For the PredictiveApriori algorithm the only parameter available is the number of rules the user wants to generate.

- As mentioned, the ARM converts the MXML log to learning instances in ARFF format. The plug-in also offers the user the option to save these learning instances as a separate ARFF file. This file can be further used for experimenting with different data mining algorithms available in the Weka library.

- The ARM also provides the option to view the intermediate frequent itemsets from which the association rules are generated.

- Besides generating the association rules, the user can also partition the input event log on the basis of generated association rules or frequent itemsets. These partitions can be further exported (using the Export functionality in the ProM framework) to a separate log file. Various mining algorithms can be applied on these new log files. So, the ARM mining plug-in serves two purposes: it first generates frequent itemsets and association rules, and second, offers the functionality of clustering. These clusters of the whole input log file help us reduce the complexity of the process models generated using the Heuristics Miner algorithm (as now we have smaller process models representing the selected rule or frequent itemset), and at the same time provide us with the knowledge about the event log in form of rules. If we do not generate these rules, these behavioural constraints existing in form of rules exist in the event log would go undiscovered.

5.1.2 Interesting rules

An association analysis algorithm has the potential to generate a large number of association patterns, but not all of them are interesting. Many of these rules are redundant and do not convey information which has not been conveyed by other rules. Such rules should be eliminated. In the ARM we introduce an additional interestingness measure for retaining the non-redundant and interesting rules from the Apriori algorithm. In our approach, we generate rules using the existing Apriori algorithm but filter the rules obtained from it before presenting them to the user. This approach is discussed below.

Consider two association rules: L1=>R1, L2=>R2 with L1, L2 as their LHS itemsets and R1, R2 as their RHS itemsets. We discard Rule 2 iff L1 is subset of L2 and R1 is superset of R2. This is shown in the figure below:

Rules:

1. L1=> R1
2. L2=>R2

Discard Rule2 if and only if:

$$L1 \subseteq L2 \,\&\, R1 \supseteq R2$$

Figure 5.1: Criteria for retaining only the interesting and non-redundant rules

Let us take some examples to illustrate our approach. Consider following three rules:

1. P1=>B, E
2. P1=> B
3. P1=>E

It is apparent that Rule 1 contains information contained in rules 2 and 3. So, both of these rules are redundant and provide no extra information and hence can be removed. Therefore, only Rule 1 is retained.

Again consider following two rules:

1. B=>E
2. B, P1=> E

In these rules, we find that whenever the task B executes the task E also executes, so Rule 2 can be removed as it depicts that whenever B and P1 executes, E also executes. In this example, Rule 1 subsumes Rule 2, and therefore Rule 2 can be removed. Also, since in association rules the emphasis is on the presence of RHS items based on the presence of an item in LHS, so in this case as we already have the information about execution of E based on execution of task B from the first rule, therefore Rule 2 can be discarded. Moreover, Rule 1 is stronger than Rule 2 because Rule 2 states that for the activity E to occur, both the activities B and P1 should execute, whereas Rule 1 indicates that for the activity E to execute only the activity B should execute.

Our approach emphasizes the retention of non-redundant rules and therefore, the redundant rules with higher confidence are also discarded. For example, out of the two rules, one with higher confidence but duplicating information contained in a low confidence rule, our approach prefers the rule with lower confidence but non-redundant information. This approach also provides the user with the most general rule [=>a_1, …a_k] where a_1…a_k represents activities in the log. This approach acts as a filter to retain non-redundant rules and it has been implemented in the ARM plug-in for the Apriori algorithm. In the next subsection, first the parameters available in both the algorithms are explained and then the experiments with the ARM are described using Case study1.

5.1.3 Parameters: Apriori algorithm

Figure 5.2 given below shows the first screen obtained for mining an event log with the ARM. The default algorithm for mining is the Apriori algorithm, but the user can also choose the PredictiveApriori algorithm for mining. The parameters for the Apriori algorithm are given below (cf. Figure 5.2):

The parameters available for the Apriori algorithm are:

1. Population size: The user can specify the population size from which the association rules will be generated. The original Apriori algorithm takes this parameter as the *number of rules*. But since we apply our approach to retain only non-redundant rules (cf. Section 5.1.2), we accept from the user, the input in form of *population size*. This indicates how many number of rules generated initially will be used for pruning based on our approach.

2. Confidence of a rule: The parameter *confidence of a rule* holds the same meaning as defined in the Section 4.1.1. Using this parameter the user can specify his desired confidence value. The default value is 0.9. The values that the user specifies is the relative value i.e. if the user specifies 0.7, it means he wants the confidence of the rule to be 70% or more.

3. Lower bound & upper bound for minimum support of an itemset: The user can also specify the support of an itemset using two parameters: *lower bound for minimum support* and *upper bound for minimum support.* By using this range of support values we can experiment with different rules that have itemsets with a support count lying in this range of values. For example, if the lower bound value is set to 0.6 and the upper bound value is set to 0.9, it means we are interested in itemsets that occur in not less than 60% of the process instances and in not more than 90% of the process instances out of the total number of process instances in the log. The default values of the lower bound and upper bound for minimum support is 0.1 and 1 respectively.

4. Output frequent itemsets: The parameter *output frequent itemsets* can be used if the user is interested in looking at the frequent itemsets too. If it is not used the output of the plug-in is only the association rules.

5. Save the intermediate ARFF: As already mentioned in the features of the plug-in in Section 5.1.2 the user can also save the intermediate learning instances in ARFF format as a separate file. This can be done using the option *save the intermediate ARFF file*.

6. Event type care information: Through this option, the ARM also provides the user the choice to work with or without the event types of activities.

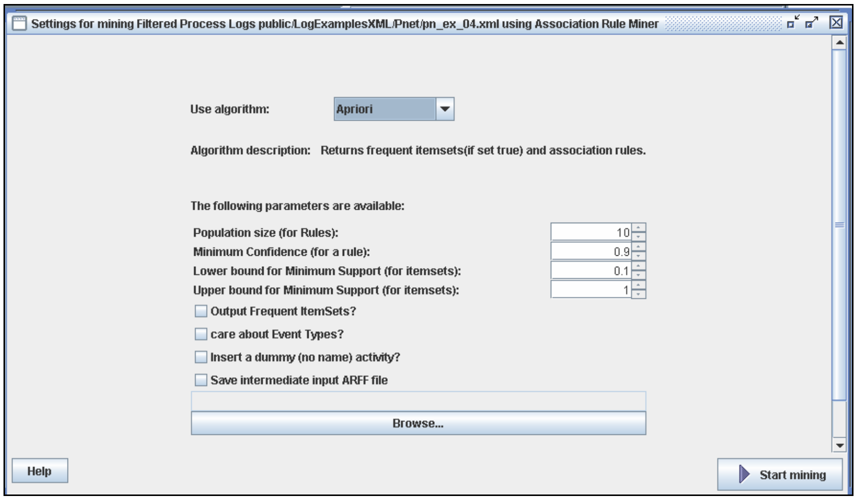
7. Insert a dummy (noname) activity: General rules of the form [=>a_1, …a_k] are generated if the user selects this option.

5.1.4 Parameters: PredictiveApriori algorithm

The parameters available in the PredictiveApriori algorithm are given below (cf. Figure 5.3):

- Number of rules: The user can specify the number of rules to be generated. The algorithm is formulated in such a way such that it returns a fixed number of best association rules rather than all rules the utility of which exceeds a given threshold. This is appropriate in many situations because a threshold may not be easy to specify and a user may not be satisfied with either an empty or an outrageously large set of rules [24].

- Save the intermediate ARFF & Event type care information: Like the Apriori algorithm, the user can also save the intermediate ARFF files and can retain event type information.



Settings for mining Filtered Process Logs public/LogExamplesXML/Pnet/pn_ex_04.xml using Association Rule Miner

Use algorithm: Apriori

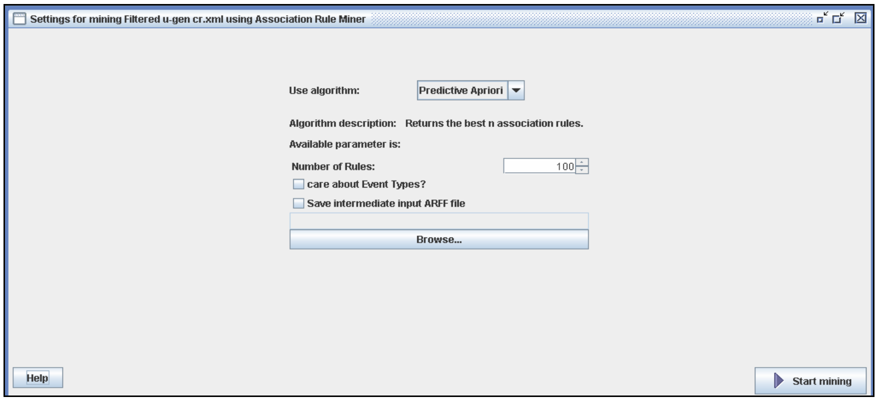
Algorithm description: Returns frequent itemsets(if set true) and association rules.

The following parameters are available:

Population size (for Rules): 10
Minimum Confidence (for a rule): 0.9
Lower bound for Minimum Support (for itemsets): 0.1
Upper bound for Minimum Support (for itemsets): 1

☐ Output Frequent ItemSets?
☐ care about Event Types?
☐ Insert a dummy (no name) activity?
☐ Save intermediate input ARFF file

Figure 5.2: The main screen of the ARM plug-in showing the parameters for the Apriori algorithm



Settings for mining Filtered u-gen cr.xml using Association Rule Miner

Use algorithm: Predictive Apriori

Algorithm description: Returns the best n association rules.

Available parameter is:

Number of Rules: 100

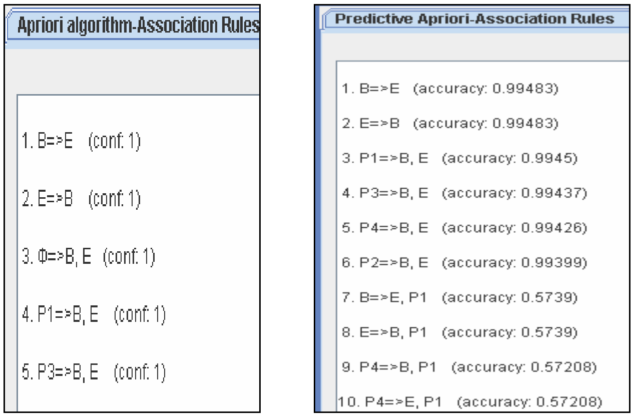
☐ care about Event Types?
☐ Save intermediate input ARFF file

Figure 5.3: PredictiveApriori algorithm in the ProM framework

Experiments with the newly implemented ARM plug-in are given in the next section.

5.2 Experimenting with the ARM

Before illustrating the experiments with the healthcare data, the output of the Apriori and the PredictiveApriori algorithms is explained using a simple log. This example log consists of 100 PIs and 6 different ATEs. Association rules obtained from both the algorithms for this event log are shown in Figure 5.4 respectively:



Apriori algorithm-Association Rules

1. B=>E (conf: 1)
2. E=>B (conf: 1)
3. Φ=>B, E (conf: 1)
4. P1=>B, E (conf: 1)
5. P3=>B, E (conf: 1)

Predictive Apriori-Association Rules

1. B=>E (accuracy: 0.99483)
2. E=>B (accuracy: 0.99483)
3. P1=>B, E (accuracy: 0.9945)
4. P3=>B, E (accuracy: 0.99437)
5. P4=>B, E (accuracy: 0.99426)
6. P2=>B, E (accuracy: 0.99399)
7. B=>E, P1 (accuracy: 0.5739)
8. E=>B, P1 (accuracy: 0.5739)
9. P4=>B, P1 (accuracy: 0.57208)
10. P4=>E, P1 (accuracy: 0.57208)

Figure 5.4: Association rules from the Apriori and the Predictive Apriori algorithms in ProM

Consider one of the association rules from the Apriori algorithm:

P1=>B, E (confidence: 1)

The rule states that if the task P1 is executed then the tasks B and E are also always executed. The confidence of the rule is 1 i.e., in all the PIs where P1 appears, the tasks B and E also appears. The rule Φ=>B, E is the most general rule and of the form [=>a_1, …a_k] as discussed in Section 5.1.2. This indicates that the activities B and E always occur together.

Figure 5.4 also shows 10 association rules from the PredictiveApriori algorithm. The interpretation of association rules generated by the PredictiveApriori algorithm is same as the interpretation of the rules computed by the Apriori algorithm. But the strength of the rule in this case is indicated by the measure: *predictive accuracy* and not by confidence as in the Apriori algorithm. PredictiveApriori algorithm computes the confidence and support values without taking them from the user and uses these values to determine the predictive accuracy of an association rule. Consider the rule:

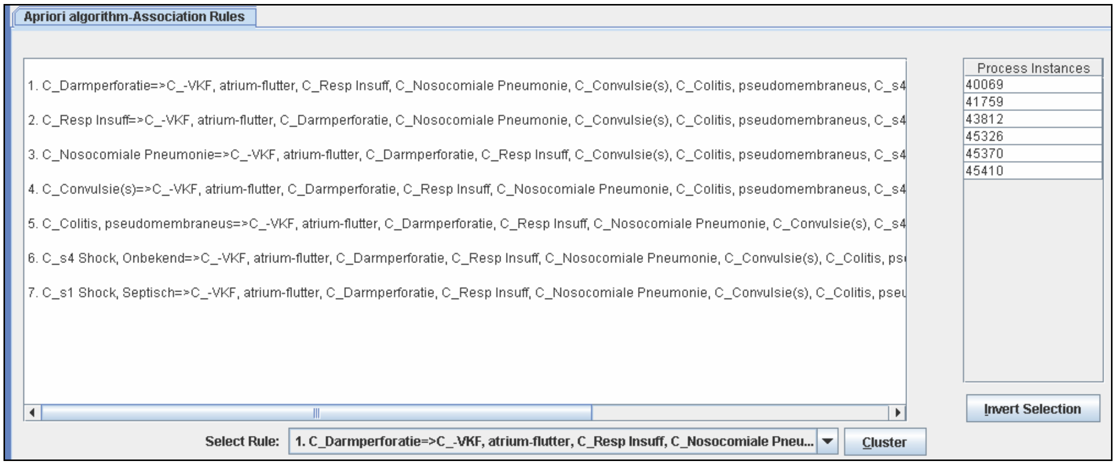
B=>E (accuracy: 0.99483)

The rule states that the task E will always execute if the task B executes and the predictive accuracy of the rule is 0.99483. Whereas, the rule B=>E, P1 states that the task E and P1 both will tend to execute if the task B is executed. The predictive accuracy of this rule is 0.5739 which is apparently lower than the predictive accuracy of the former rule indicating that the certainty of the latter rule in future is lesser than that of the former rule.

In the next subsections, experiments with healthcare data from Case study1 are illustrated.

5.2.1 Illustration 1

The log used in this experiment pertains to patients suffering from the complications of type 'uro-genitaal'. It contains 6 PIs and 18 ATEs. The process model for this log can be found in Chapter 3, Figure 3.13. The result of applying association rule mining with default parameter settings of the Apriori algorithm are shown in Figure 5.5. It can be seen that 7 interesting rules (with confidence 1) are retained from population size of 10.



Apriori algorithm-Association Rules

1. C_Darmperforatie=>C_-VKF, atrium-flutter, C_Resp Insuff, C_Nosocomiale Pneumonie, C_Convulsie(s), C_Colitis, pseudomembraneus, C_s4

2. C_Resp Insuff=>C_-VKF, atrium-flutter, C_Darmperforatie, C_Nosocomiale Pneumonie, C_Convulsie(s), C_Colitis, pseudomembraneus, C_s4

3. C_Nosocomiale Pneumonie=>C_-VKF, atrium-flutter, C_Darmperforatie, C_Resp Insuff, C_Convulsie(s), C_Colitis, pseudomembraneus, C_s4

4. C_Convulsie(s)=>C_-VKF, atrium-flutter, C_Darmperforatie, C_Resp Insuff, C_Nosocomiale Pneumonie, C_Colitis, pseudomembraneus, C_s4

5. C_Colitis, pseudomembraneus=>C_-VKF, atrium-flutter, C_Darmperforatie, C_Resp Insuff, C_Nosocomiale Pneumonie, C_Convulsie(s), C_s4

6. C_s4 Shock, Onbekend=>C_-VKF, atrium-flutter, C_Darmperforatie, C_Resp Insuff, C_Nosocomiale Pneumonie, C_Convulsie(s), C_Colitis, ps

7. C_s1 Shock, Septisch=>C_-VKF, atrium-flutter, C_Darmperforatie, C_Resp Insuff, C_Nosocomiale Pneumonie, C_Convulsie(s), C_Colitis, pseu

Process Instances
40069
41759
43812
45326
45370
45410

Select Rule: 1. C_Darmperforatie=>C_-VKF, atrium-flutter, C_Resp Insuff, C_Nosocomiale Pneu...

Figure 5.5: Association rules indicating careflows for patients suffering with "uro-genitaal" type complications

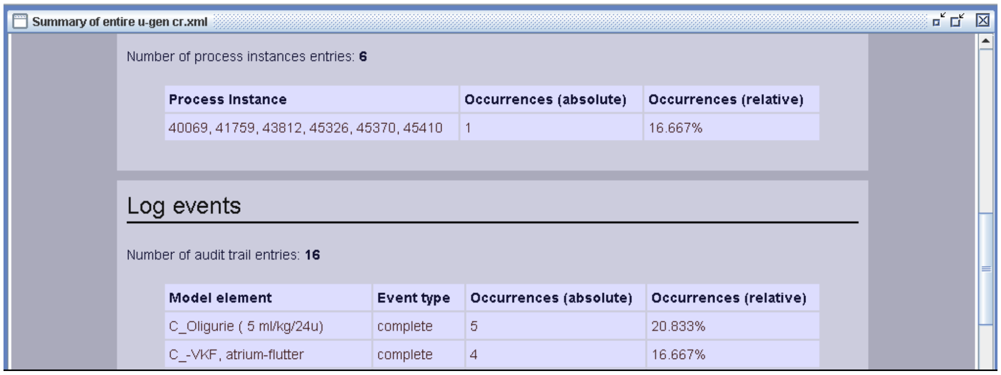
Let us analyze one of these rules shown in Figure 5.5:

C_Nosocomiale Pneumonie=>C_-VKF, atrium-flutter, C_Darmperforatie, C_Resp Insuff, C_Convulsie(s), C_Colitis, pseudomembraneus, C_s4 Shock, Onbekend, C_s1 Shock, Septisch, C_Oligurie (< 5 ml/kg/24u)

According to this rule, if a patient suffers from complication *C_Nosocomiale Pneumonie*, he also tends to suffer from other complications listed on the RHS of the rule. A confidence of 1 indicates that 100% of the patients (whose data is recorded in this event log) who suffered from this complication also suffered from complications given on the RHS. It can be seen from the process model that the task C_-*VKF, atrium-flutter* can be followed by the tasks *C_-SVT, paroxysmaal* and *C_s4 Shock, Onbekend.* But a patient suffering from *C_Nosocomiale Pneumonie* always follows the path indicated by the task *C_s4 Shock, Onbekend* and not *C_-SVT, paroxysmaal*.

It is also observed that all the rules generated in this experiment consist of 9 items (complication activities). These are: *C_Nosocomiale Pneumonie*, *C_-VKF, atrium-flutter*, *C_Darmperforatie*, *C_Resp Insuff*, *C_Convulsie(s)*, *C_Colitis, pseudomembraneus*, *C_s4 Shock, Onbekend*, *C_s1 Shock, Septisch*, and *C_Oligurie (< 5 ml/kg/24u).* Looking at these rules it can be said that any patient suffering from any of these complications except *C_-VKF, atrium-flutter* and *C_Oligurie (< 5 ml/*kg/24u) also suffers from all other complications in this list. But there is no association rule indicating that a patient suffering from any of these two complications suffers from other listed complications too.

When this experiment was re-performed with a larger population size, a rule indicating that any patient who suffers from both of these complications *C_-VKF, atrium-flutter* and *C_Oligurie (< 5 ml/*kg/24u) suffers from the other listed complications was retrieved. This shows that a patient always suffer from these two complications at the same time. This rule is also verified by looking at the log summary in ProM (cf. Figure 5.6). It can be seen that these two complications are the most frequently occurring complications in the log. However, the importance of this rule can be correctly understood by any person with adequate domain knowledge. But it is sure that these rules can help the healthcare organizations improve their services by making anticipatory preparations based on the information contained in the rule.



Number of process instances entries: **6**

Process Instance	Occurrences (absolute)	Occurrences (relative)
40069, 41759, 43812, 45326, 45370, 45410	1	16.667%

Log events

Number of audit trail entries: **16**

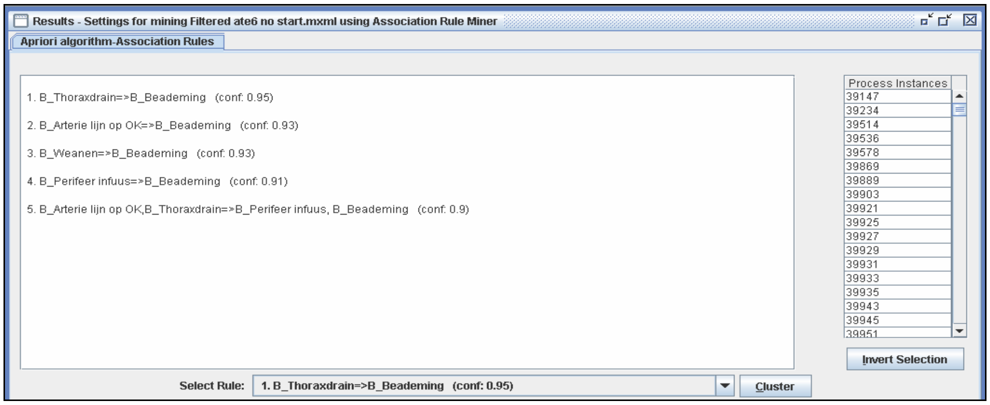
Model element	Event type	Occurrences (absolute)	Occurrences (relative)
C_Oligurie (5 ml/kg/24u)	complete	5	20.833%
C_-VKF, atrium-flutter	complete	4	16.667%

Figure 5.6: The 2 most frequently occurring complications: *C_Oligurie* and *C_VKF* always happen together as stated by an association rule

5.2.2 Illustration 2

In this experiment the ARM was applied to a hospital log consisting of treatment and some complication activities. The log has 2269 cases and 174 different ATEs. The Apriori algorithm was applied with default parameter settings except the population size which was set to 100. The rules obtained are shown in Figure 5.7. Though the population size was set to 100, we obtained only 5 rules because a lot of low frequent events are found in the log. From a total of 174 ATEs, 143 ATEs occur less than 50 times and, 165 ATEs occur less than 100 times. Because this experiment was done with high values of support threshold therefore such low frequent behaviour was not captured in form of association rules.

This experiment gave us insights into the underlying process. The presence of a lot of low frequent behaviour may indicate the presence of noise. However, the degree of noise is unknown. In this experiment, the lower bound minimum support value was set to 0.1 which indicates that we are interested in rules that involve activities occurring not less than 10% in the log. But this value of lower bound for minimum support is also a high value for the activities with very low frequency and hence the algorithm could not generate association rules involving a lot of activities.



Results - Settings for mining Filtered ate6 no start.mxml using Association Rule Miner

Apriori algorithm-Association Rules

1. B_Thoraxdrain=>B_Beademing (conf: 0.95)

2. B_Arterie lijn op OK=>B_Beademing (conf: 0.93)

3. B_Weanen=>B_Beademing (conf: 0.93)

4. B_Perifeer infuus=>B_Beademing (conf: 0.91)

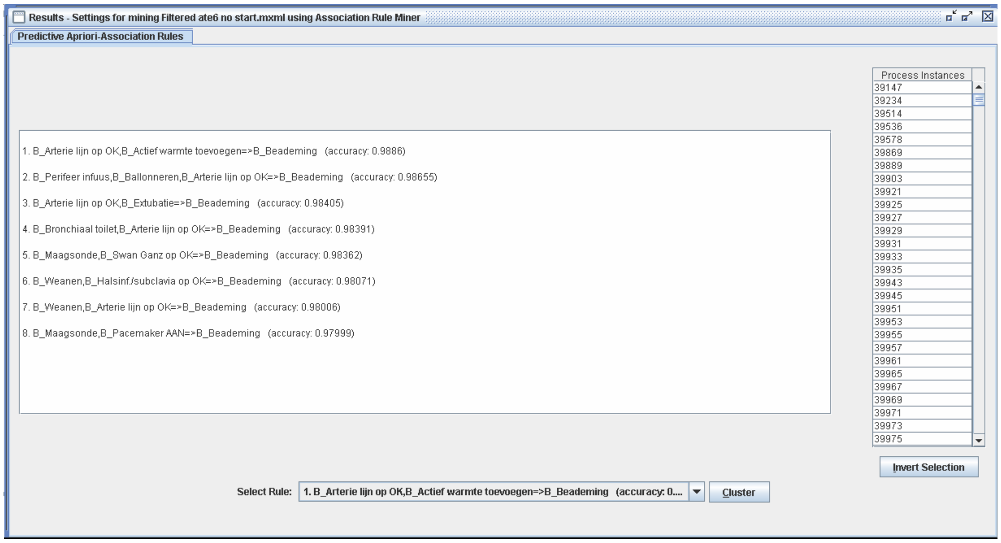
5. B_Arterie lijn op OK,B_Thoraxdrain=>B_Perifeer infuus, B_Beademing (conf: 0.9)

Process Instances
39147
39234
39514
39536
39578
39869
39889
39903
39921
39925
39927
39929
39931
39933
39935
39943
39945
39951

Select Rule: 1. B_Thoraxdrain=>B_Beademing (conf: 0.95)

Figure 5.7: High support threshold for association rules do not capture low frequent behaviour

The PredictiveApriori algorithm was also applied to this log and the resulting rules are seen in the Figure 5.8. It should be noted that this algorithm captures greater number of activities even with 8 rules (desired number of rules =10) as compared to the 5 rules by the Apriori algorithm in Figure 5.7. But rules from both the algorithms indicate that the activity *B_Beademing* is the most frequent treatment activity. The rules in the following figure indicate the treatment paths followed by different patients who eventually receive the treatment *B_Beademing*.



Results - Settings for mining Filtered ate6 no start.mxml using Association Rule Miner

Predictive Apriori-Association Rules

1. B_Arterie lijn op OK,B_Actief warmte toevoegen=>B_Beademing (accuracy: 0.9886)

2. B_Perifeer infuus,B_Ballonneren,B_Arterie lijn op OK=>B_Beademing (accuracy: 0.98655)

3. B_Arterie lijn op OK,B_Extubatie=>B_Beademing (accuracy: 0.98405)

4. B_Bronchiaal toilet,B_Arterie lijn op OK=>B_Beademing (accuracy: 0.98391)

5. B_Maagsonde,B_Swan Ganz op OK=>B_Beademing (accuracy: 0.98362)

6. B_Weanen,B_Halsinf./subclavia op OK=>B_Beademing (accuracy: 0.98071)

7. B_Weanen,B_Arterie lijn op OK=>B_Beademing (accuracy: 0.98006)

8. B_Maagsonde,B_Pacemaker AAN=>B_Beademing (accuracy: 0.97999)

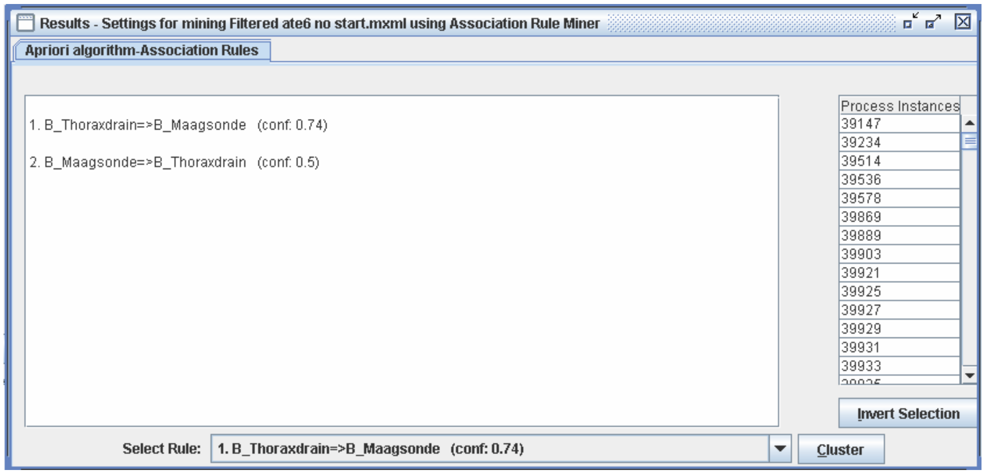
Process Instances
39147
39234
39514
39536
39578
39869
39889
39903
39921
39925
39927
39929
39931
39933
39935
39943
39945
39951
39953
39955
39957
39961
39965
39967
39969
39971
39973
39975

Select Rule: 1. B_Arterie lijn op OK,B_Actief warmte toevoegen=>B_Beademing (accuracy: 0....

Figure 5.8: Association rules from the PredictiveApriori algorithm

5.2.3 Illustration 3

The event log used in Illustration 2 is also used in this experiment. The values of different parameters for the Apriori algorithm are: population size=10, confidence=0.5, lower bound for minimum support=0.1 and upper bound for minimum support=0.5. This resulted in generation of 2 association rules as seen in Figure 5.9. These rules are of the form: a=>b and b=>a, with confidence of 0.74 and 0.5 respectively. From these rules it can be interpreted that 74% of the patients from the total patients whose data is recorded in this event log receiving the treatment *B_Thoraxdraine* also receives the treatment *B_Maagsonde*, whereas only 50% of the patients who are first given the treatment *B*_Maagsonde receive the treatment *B*_Thoraxdraine.



Results - Settings for mining Filtered ate6 no start.mxml using Association Rule Miner

1. B_Thoraxdrain=>B_Maagsonde (conf: 0.74)

2. B_Maagsonde=>B_Thoraxdrain (conf: 0.5)

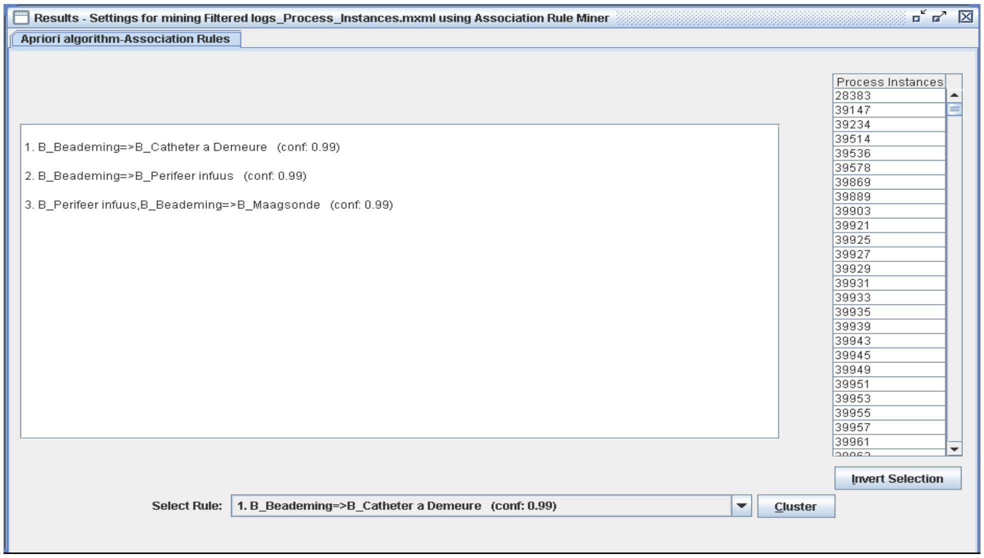
Process Instances
39147
39234
39514
39536
39578
39869
39889
39903
39921
39925
39927
39929
39931
39933

Select Rule: 1. B_Thoraxdrain=>B_Maagsonde (conf: 0.74)

Figure 5.9: Association rules with highly frequent log activities

5.2.4 Illustration 4

For this experiment, the same treatment log as used for Figure 3.12 (Chapter 3) is used. The result of mining this log with the Apriori algorithm (default settings) is given in the figure below. Three association rules are obtained, each with a confidence of 0.99.



Results - Settings for mining Filtered logs_Process_Instances.mxml using Association Rule Miner

1. B_Beademing=>B_Catheter a Demeure (conf: 0.99)

2. B_Beademing=>B_Perifeer infuus (conf: 0.99)

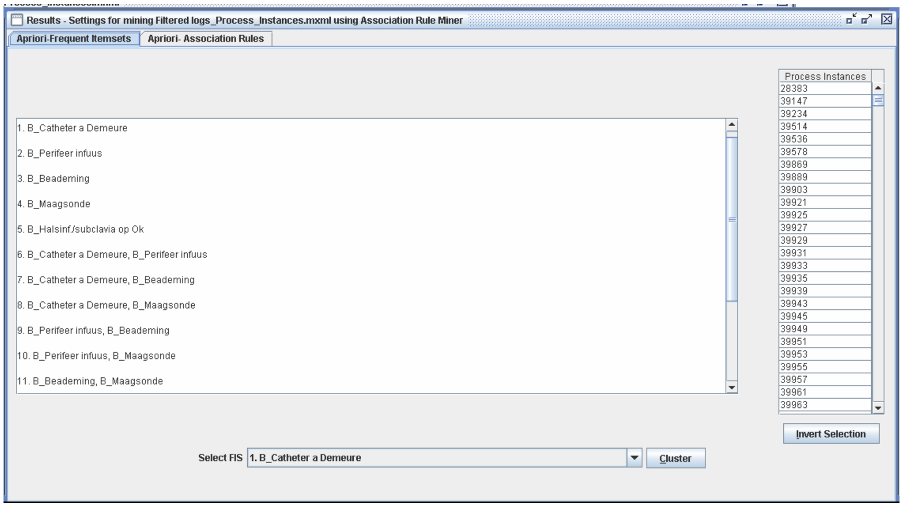
3. B_Perifeer infuus,B_Beademing=>B_Maagsonde (conf: 0.99)

Process Instances
28383
39147
39234
39514
39536
39578
39869
39889
39903
39921
39925
39927
39929
39931
39933
39935
39939
39943
39945
39949
39951
39953
39955
39957
39961

Select Rule: 1. B_Beademing=>B_Catheter a Demeure (conf: 0.99)

Figure 5.10: Association rules with highly frequent log activities

The frequent itemsets for this log can also be retrieved using the option *output frequent itemsets*. These itemsets can be seen in Figure 5.11. This figure shows the set of items that together appear frequently, for example, *B_Catheter a Demeure, B_Perifeer* infuus is a set of treatment activities that always appear frequently together. When we have the support count of this frequent itemset, we can interpret that 'n' number of patients (where n is the support count) always receive these treatments together. This support count can be retrieved using the ARM plug-in. This is illustrated in the next chapter.


Results - Settings for mining Filtered logs_Process_Instances.mxml using Association Rule Miner
Apriori-Frequent Itemsets
Apriori- Association Rules
Process Instances
Invert Selection
Select FIS
Cluster

1. B_Catheter a Demeure
2. B_Perifeer infuus
3. B_Beademing
4. B_Maagsonde
5. B_Halsinf./subclavia op Ok
6. B_Catheter a Demeure, B_Perifeer infuus
7. B_Catheter a Demeure, B_Beademing
8. B_Catheter a Demeure, B_Maagsonde
9. B_Perifeer infuus, B_Beademing
10. B_Perifeer infuus, B_Maagsonde
11. B_Beademing, B_Maagsonde

Process Instances
28383
39147
39234
39514
39536
39578
39869
39889
39903
39921
39925
39927
39929
39931
39933
39935
39939
39943
39945
39949
39951
39953
39955
39957
39961
39963

Select FIS: 1. B_Catheter a Demeure

Figure 5.11: Frequent itemsets for a treatments log activities

The next section describes how different factors like number of PIs, number of ATEs can affect the execution time of these algorithms.

5.3 Performance issues

To evaluate the performance of the association rule algorithms in terms of computational time we conducted some tests. For the first test, the number of process instances in a healthcare log was increased 10 times, and the time taken by the Apriori and PredictiveApriori algorithms was recorded. For the second test, the number of ATEs i.e. events in an event log was gradually decreased and the effect on execution time of the two algorithms was recorded and plotted. These tests were conducted on Intel(R) Pentium(R) 4 CPU, with 3.40GHz, 1.99 GB of RAM. These tests are described below

- **Number of PIs vs. Computation time:** Table 5.1 gives the 10 logs used for studying the relation between number of PIs of an event log and the time taken by the Apriori and the PredictiveApriori algorithms in generating association rules. The initial log (Log_1) contained 38 PIs and 65 different ATEs. The number of PIs of this log was increased up to 10 times. Table 5.1 gives details of these logs. Before presenting the results, first the dimensions of these logs are given in terms of number of PIs and number of events (ATEs), both number of different ATEs and the total number of ATEs.

Table 5.1: Log profiles

Log name	#PIs	# different ATEs	Total # ATEs
Log_1	38	65	273
Log_2	76	65	546
Log_3	114	65	819
Log_4	152	65	1092
Log_5	190	65	1365
Log_6	228	65	1638
Log_7	266	65	1911
Log_8	304	65	2184
Log_9	342	65	2457
Log_10	380	65	2730

It should be noted that when the number of PIs in the log is increased, number of total ATEs also increases. Table 5.2 gives the computation time of the two algorithms for different number of PIs. The data given in the Table 5.2 is plotted as graphs and shown in Figure 5.12.

Table 5.2: Number of PIs and corresponding algorithm computation times

Log name	#PIs	Computation time (milliseconds)	
		Apriori	PredictiveApriori
Log_1	38	47	2046
Log_2	76	63	2937
Log_3	114	78	3922
Log_4	152	94	4875
Log_5	190	125	5750
Log_6	228	140	6718
Log_7	266	156	8000
Log_8	304	172	8328
Log_9	342	188	9375
Log_10	380	203	10281

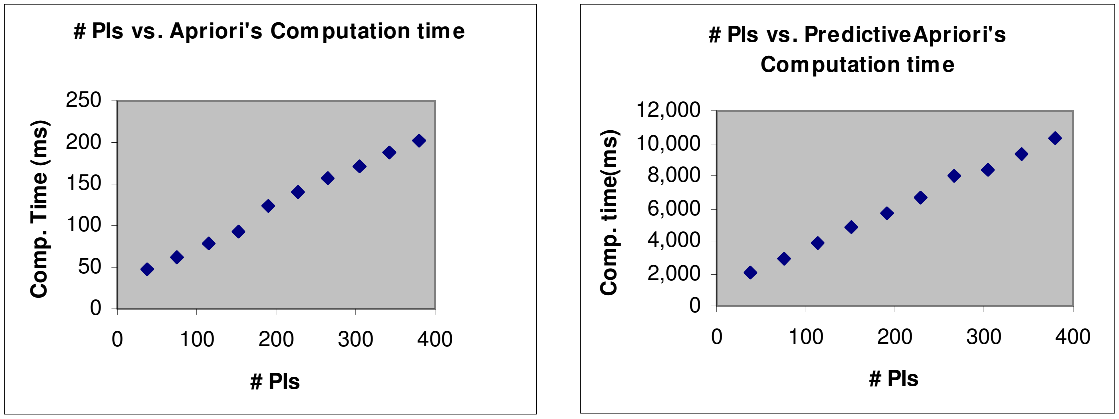


Figure 5.12: Number of PIs vs. Computation time: Apriori and PredictiveApriori algorithms

Figure 5.12 shows that for both the algorithms the computation time increases with an increase in number of process instances in the event log. The Apriori algorithm makes repeated passes over the event log therefore its run time increases with the size of the event log. The PredictiveApriori algorithm however consumes more time compared to the Apriori algorithm, as can also be seen from the above figure. For the same increase in number of PIs, the computation time of the PredictiveApriori increases more than 5 times. It should be noted that in general also, the PredictiveApriori takes more time than the Apriori algorithm (cf. computation time for Log_1). Figure 5.13 shows the comparison between the Apriori and the PredictiveApriori algorithms in terms of computation time.

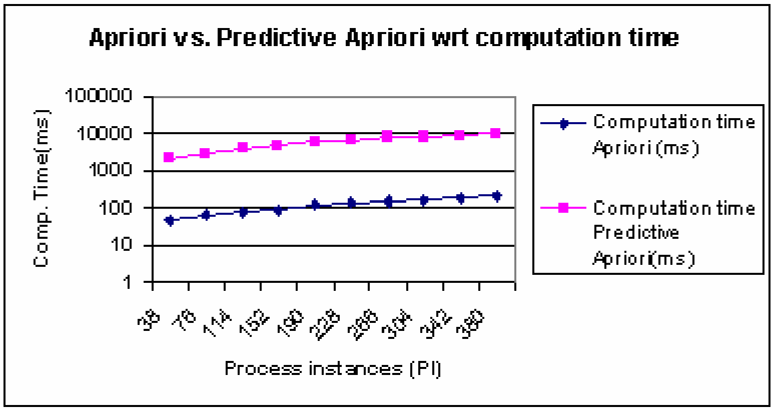


Figure 5.13: Apriori vs. PredictiveApriori wrt computation time

- **Number of ATEs vs. Computation time:** In this experiment, the number of process instances are kept constant and the number of events i.e. number of ATEs in the log are varied. It should be noted that when we vary the number of ATEs in the log, the structure of the log changes as both the number of ATEs as well as the total number of ATEs changes. The total number of ATEs also decreases as the number of different ATEs is decreased and this affects the computation time. From the above experimental setup the healthcare log with maximum number of PIs i.e. the log with 380 PIs and 65 ATEs was taken as the initial log, and at a time 10 randomly chosen ATEs were removed from this log till the log has substantial number of ATEs. Table 5.3 gives the computation time of the Apriori and PredictiveApriori algorithms for the reduced number of ATEs. The data given in this table is plotted as graphs and shown in Figure 5.14.

Table 5.3: Number of ATEs and corresponding algorithm computation times

Log name	#different ATEs	Total #ATEs	Computation time (milliseconds)	
			Apriori	PredictiveApriori
Log_1	65	2730	188	10328
Log_2	55	2130	172	7046
Log_3	45	1780	125	6219
Log_4	35	1500	93	4266
Log_5	25	1300	47	2844
Log_6	15	1050	31	1578

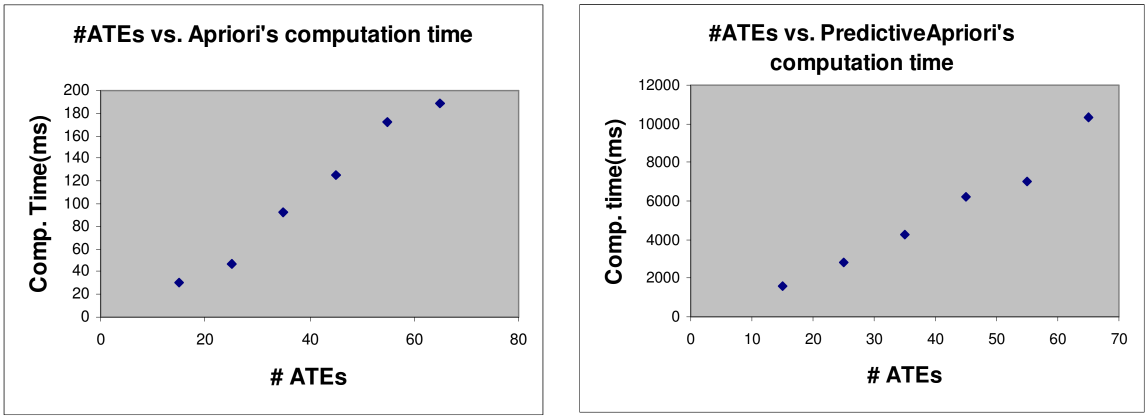


Figure 5.14 Number of ATEs vs. Computation time: Apriori and PredictiveApriori algorithm

In both the algorithms, as the number of ATEs increases, the number of items declared as frequent also increases. This increase in the number of events increases the computation time and I/O costs as larger number of candidate itemsets will be generated by the algorithms. For the same reason, when the number of events is decreased from 65 to 15, the computation time of both the algorithms decreased because now less number of events are declared to be frequent. However, the execution time of the PredictiveApriori drops more than as compared to that of the Apriori algorithm.

From the above experiments, it is apparent that the Apriori algorithm is a faster algorithm as compared to the PredictiveApriori algorithm. Computation time of both the algorithms increases linearly with an increase in size of the event log in terms of PIs. Also, the computation time of both the algorithms reduces with a decrease in width of an event log in terms of ATEs. It should be noted that these experiments were conducted on a computer with large amount of memory (1.99GB RAM) and high CPU speed. Performance of these algorithms will drop down with lesser amounts of memory.

5.4 Conclusion

In this chapter newly implemented ARM plug-in was presented. The chapter described the salient features of the ARM. As it is known that association analysis algorithms can produce lots of rules but not all of them are useful. Therefore in Section 5.2 we suggested an approach for obtaining non-redundant rules. The chapter also illustrates some experiments done on Case study1 in order to achieve the third research goal of this thesis. The experimental results favour the use of association rules for mining healthcare processes. It was seen that the association rules are easy to understand (as compared to the discriminant rules) and present behavioural patterns implicit in the log. These rules can be used for obtaining groups of similar patients. The clustering functionality that makes it possible is the subject of the next chapter.

6 Clustering

In Chapter 4, association rules were found to be a promising technique to obtain information about any process underlying an event log or to make implicit knowledge explicit. To further explore this research area, the ARM was proposed and implemented in the ProM framework. Section 5.2 presented the experimental results of using the Case study1 and the ARM. It was seen that the Case study1 includes a lot of low frequent events. Many events have frequency of 1 whereas the highest frequency of an activity in the log is in 1000s. It was seen that the ARM plug-in easily captures high frequent behaviour whereas the low frequent behaviour remains dependent on the support threshold values. The ARM however can not distinguish noise with the low frequent behaviour. The presence of highly low frequent behaviour represents flexibility and lack of standardization in the Case study1 or it may represent rare and specialized medical cases pertaining to a unique genre of patients.

The low frequent behaviour in any healthcare log adds up to the heterogeneity of the data as it represents more flexibility, uniqueness and less structure. It is also a kind of behavioural pattern that exists in the log and often goes unnoticed and undiscovered. However, whether some behaviour is low frequent or high frequent if there would be a mechanism to group patients with similar profile, it would present an opportunity to obtain homogeneity in the heterogeneous and less-structured healthcare processes. For this purpose, the ARM provides the functionality of clustering. As apparent, this chapter focuses on the fourth research goal:

Develop a mechanism to use Association Rules for clustering different patient (or complications, treatments etc.) groups into one homogeneous group.

The chapter begins with a brief introduction about clustering in Section 6.1. Section 6.2 is the experimental section where some examples from Case study1 illustrate how clustering can be achieved in the ARM plug-in. Section 6.3 concludes this chapter by summarizing the importance of clustering in combination with the association rules.

6.1 Introduction

In most simple terms, clustering can be understood as making clusters of similar things. It is the process of organizing objects into groups whose members are similar in some way. A cluster is therefore a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters [32]. In the process of clustering the aim is to find homogeneous chunks in data.

Clustering is a key area in data mining and knowledge discovery, which are activities oriented towards finding non-trivial or hidden patterns in data collected in databases. Commonly, the applications of clustering includes finding common surfing patterns in the set of web users, partitioning different documents based on their content, finding protein sequences in a database, finding groups of customers with similar purchase patterns etc. All these applications aim at finding homogeneous members in a database, and these groups can be further used as a target for understanding behaviour of elements (like customers in a supermarket, relationship between documents, protein sequences etc.) so that insights can be gained into the patterns that exist in a pool of data. Clustering process instances in an event log also aim at finding homogeneity in the log.

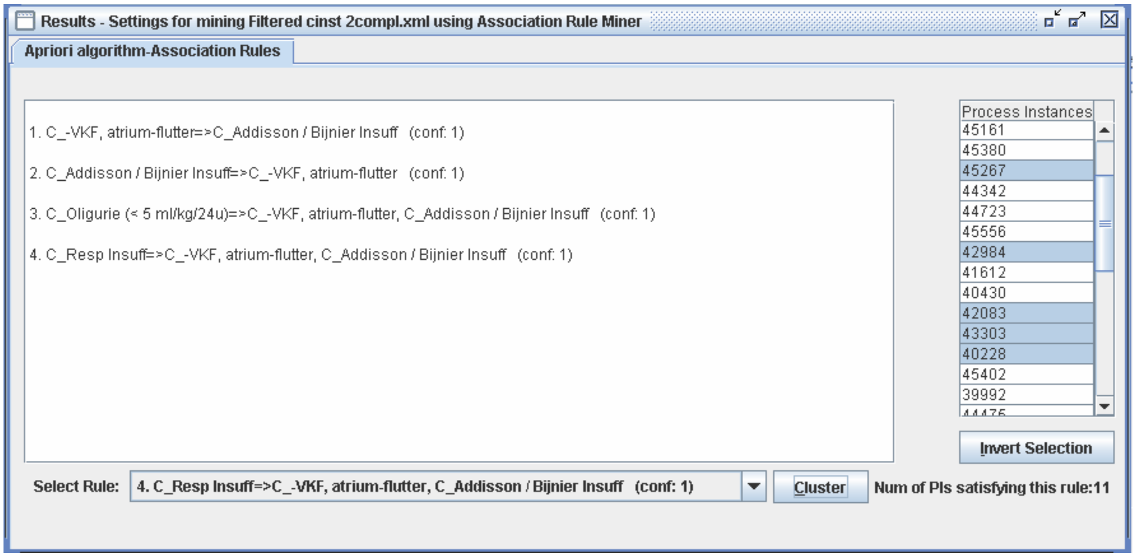
In healthcare domain, processes are dynamic, less structured and involves various disciplines. Every patient represents unique case in terms of complications, body type, responses to treatment procedures etc. In this situation it becomes difficult to find a group of patients that are similar in one way or another. Clustering in context of ARM refers to finding homogeneous groups of process instances (patients) which are similar in some way. The ARM provides two choices for clustering: homogeneous group of PIs/cases can be found that satisfy certain association rule, and secondly homogeneous group of PIs/cases can be found on the basis of a particular frequent pattern in the log (itemset). Clustering in the ARM is in the form of partitioning, where the entire event log is partitioned in two clusters: one which satisfies a particular association rule/frequent itemset and the other which does not satisfy that it.

6.2 Clustering in the ARM

This section illustrates by some examples how the ARM can also be used to derive clusters of homogeneous patients from a healthcare log.

6.2.1 Illustration 1

To illustrate how clusters of similar cases can be derived using association rules as the clustering criteria, a simple example is shown below. Association rules are generated for a complications log consisting of 38 PIs and 65 different ATEs. Each PI refers to a patient and the complication path followed by him. The ARM applied with default parameter values generates 4 association rules. Observing these 4 rules, it can be interpreted, that the complications *C_-VKF atrium- flutter* and *C_Addisson/Bijnier Insuff* always occur together in this log and are the most frequently occurring complications. Therefore, we chose a rule involving both these complications as the basis for clustering. This homogeneous cluster of PIs satisfying Rule 4 can be obtained by clicking the button *cluster*. This cluster can be seen in the table at the RHS in Figure 6.1. This table lists all the PIs in the event log and the PIs satisfying this association rule are selected in the figure.



Results - Settings for mining Filtered cinst 2compl.xml using Association Rule Miner

Apriori algorithm-Association Rules

1. C_-VKF, atrium-flutter=>C_Addisson / Bijnier Insuff (conf: 1)

2. C_Addisson / Bijnier Insuff=>C_-VKF, atrium-flutter (conf: 1)

3. C_Oligurie (< 5 ml/kg/24u)=>C_-VKF, atrium-flutter, C_Addisson / Bijnier Insuff (conf: 1)

4. C_Resp Insuff=>C_-VKF, atrium-flutter, C_Addisson / Bijnier Insuff (conf: 1)

Process Instances
45161
45380
45267
44342
44723
45556
42984
41612
40430
42083
43303
40228
45402
39992

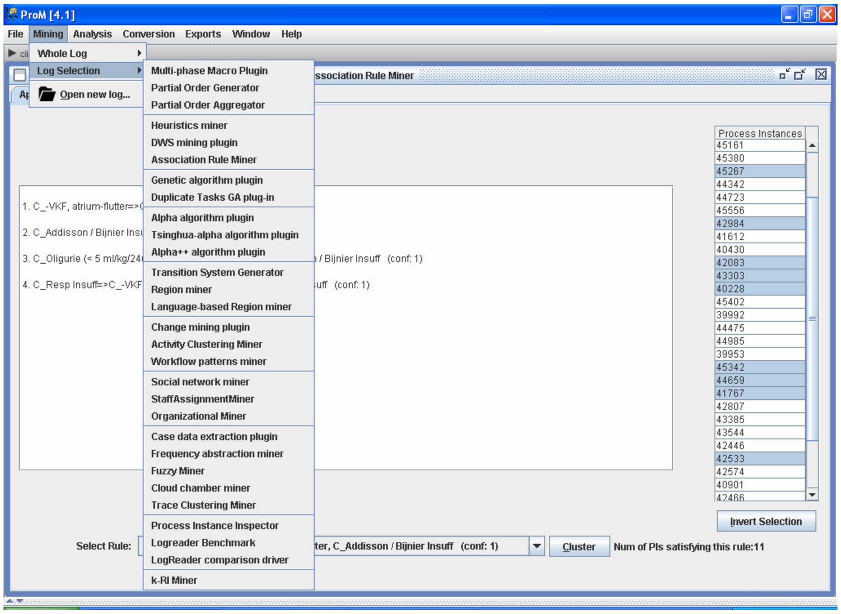
Select Rule: 4. C_Resp Insuff=>C_-VKF, atrium-flutter, C_Addisson / Bijnier Insuff (conf: 1)

Num of PIs satisfying this rule:11

Figure 6.1: Clustering in the ARM

This cluster can be further used as input for other mining algorithms like the HM, the Alpha algorithm, the Genetic Algorithm plug-in, the Fuzzy Miner, the DWS plug-in, etc. and for analysis plug-ins like the LTL checker, performance sequence diagram analysis etc. This can be seen in figures 6.2 and 6.3 respectively. Besides this, the selected PIs can also be inverted and the cluster of PIs not satisfying the particular rule can be obtained.

When this cluster is used as an input for the HM algorithm, the process model obtained is much simpler than the process model obtained for the complete log. This represents the usefulness of the clustering functionality. It helps in obtaining simpler (as compared to the spaghetti-models) models which provide better insights into the underlying process. In Figure 6.4 we present the process models (only the structure) corresponding to the entire log and cluster representing Rule 4 respectively. The process model for Rule 4 is simpler than the process model for the entire log. Though the problems like dangling activities and missing connections exist in these process models, these may be eliminated by accordingly varying the parameter settings of the HM algorithm so as to generate only the detailed behaviour and leave the low frequent one. This process model gives us information about the control flow of the patients suffering from the complication *C_Resp Insuff* that eventually also suffer from the complications *C_-VKF atrium- flutter* and *C_Addisson/Bijnier Insuff.*



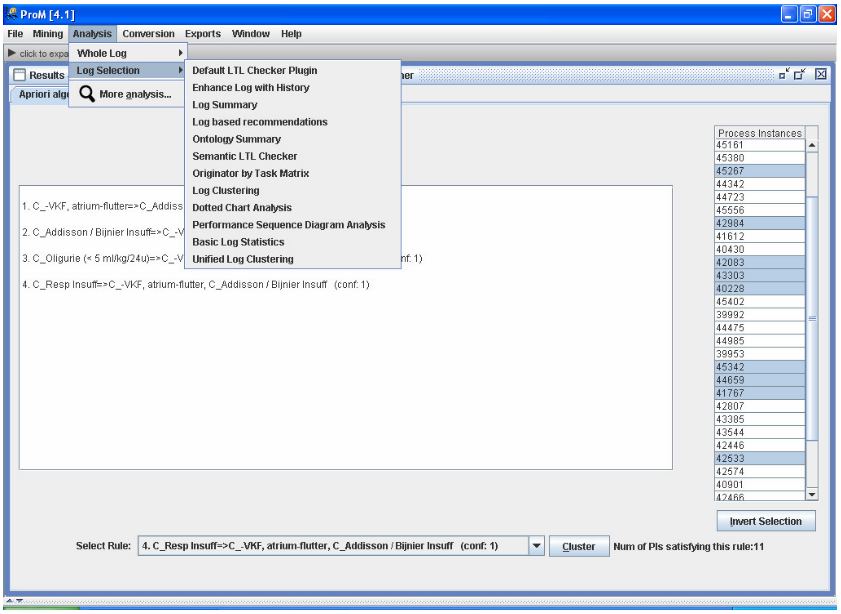
1. C_-VKF, atrium-flutter=>(
2. C_Addisson / Bijnier Insu
3. C_Oligurie (< 5 ml/kg/24 / Bijnier Insuff (conf: 1)
4. C_Resp Insuff=>C_-VKF uff (conf: 1)

Process Instances: 45161, 45380, 45267, 44342, 44723, 45556, 42984, 41612, 40430, 42083, 43303, 40228, 45402, 39992, 44475, 44985, 39953, 45342, 44659, 41767, 42807, 43385, 43544, 42446, 42533, 42574, 40901, 42466

Select Rule: ter, C_Addisson / Bijnier Insuff (conf: 1)

Num of PIs satisfying this rule:11

Figure 6.2: Clustered PIs can be used for other mining algorithms



1. C_-VKF, atrium-flutter=>C_Addiss
2. C_Addisson / Bijnier Insuff=>C_-V
3. C_Oligurie (< 5 ml/kg/24u)=>C_-V nf: 1)
4. C_Resp Insuff=>C_-VKF, atrium-flutter, C_Addisson / Bijnier Insuff (conf: 1)

Process Instances: 45161, 45380, 45267, 44342, 44723, 45556, 42984, 41612, 40430, 42083, 43303, 40228, 45402, 39992, 44475, 44985, 39953, 45342, 44659, 41767, 42807, 43385, 43544, 42446, 42533, 42574, 40901, 42466

Select Rule: 4. C_Resp Insuff=>C_-VKF, atrium-flutter, C_Addisson / Bijnier Insuff (conf: 1)

Num of PIs satisfying this rule:11

Figure 6.3: Clustered PIs can be used for other analysis algorithms

6.2.2 Illustration 2

This illustration describes another use of the clustering functionality provided in the ARM plug-in. Clustering can be used to obtain the support count of the antecedents and consequent itemsets in a rule. For this, the user must choose the option *Output Frequent Itemsets* parameter available in the plug-in (cf. Figure 5.2). For the log used in the above illustration, the output after choosing this option is as seen in the Figure 6.5. The two tabs seen in the figure provide information about the frequent itemsets and the association rules respectively. It is seen that one of the frequent itemsets is the itemset- *C_Oligurie (<5ml/kg/24u)* and the number of process instances in which this itemset occurs (support count of this itemset, cf. Equation 4.2) in the event log can be obtained by selecting this itemset and clicking the button *Cluster*. This gives us the process instances which satisfy this itemset, and its count. The number of PIs satisfying this FIS is 12. Similarly, the support count for the itemset *C_-VKF atrium- flutter* and *C_Addisson/Bijnier Insuff* is found to be 38 indicating that all the patients in this log suffer from these two complications. The support count of different itemsets gives an indication of the frequency of a particular complication. For example, the support count of itemset *C_-VKF atrium- flutter* and *C_Oligurie (<5ml/kg/24u)* is 12 indicating that out of 38 patients (support count of *C_-VKF atrium-* flutter) that suffer from the complication *C_-VKF atrium- flutter*, only 12 patients also suffer from *C_Oligurie (<5ml/kg/24u).* This will help the stakeholders in understanding the relative criticality of the complications (in terms of their frequency) and with this prior information they can perform their tasks efficiently in terms of time, efforts, cost and quality.

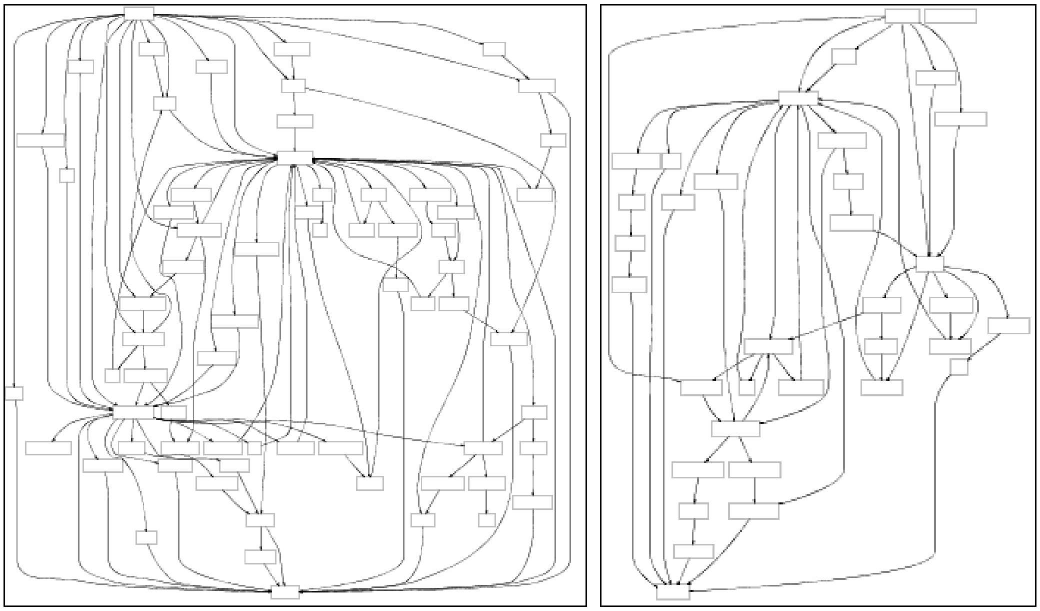
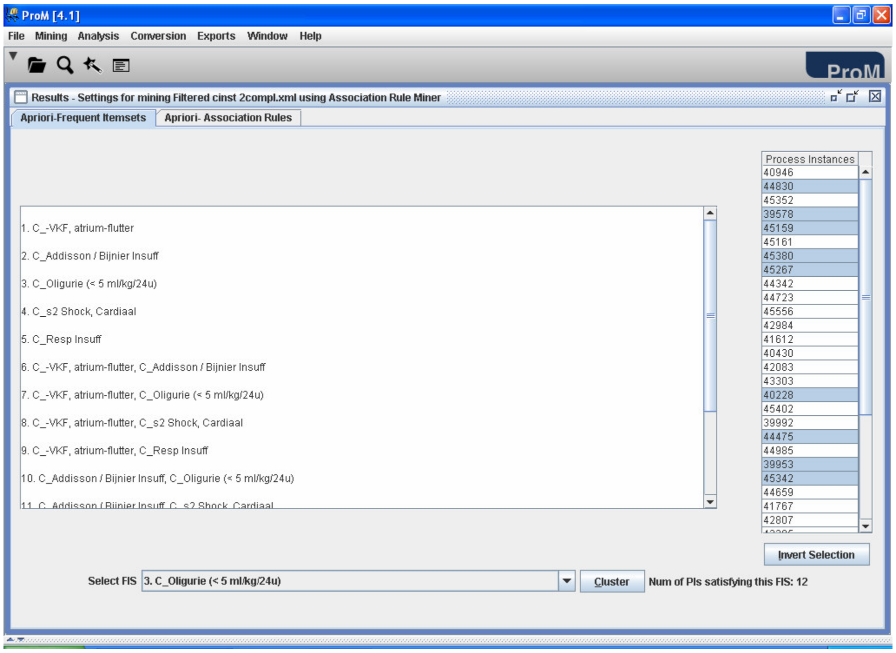


Figure 6.4: Process models for the entire log and Rule 4 (respectively)



ProM [4.1]
File Mining Analysis Conversion Exports Window Help
ProM

Results - Settings for mining Filtered cinst 2compl.xml using Association Rule Miner

Apriori-Frequent Itemsets
Apriori- Association Rules

1. C_-VKF, atrium-flutter
2. C_Addisson / Bijnier Insuff
3. C_Oligurie (< 5 ml/kg/24u)
4. C_s2 Shock, Cardiaal
5. C_Resp Insuff
6. C_-VKF, atrium-flutter, C_Addisson / Bijnier Insuff
7. C_-VKF, atrium-flutter, C_Oligurie (< 5 ml/kg/24u)
8. C_-VKF, atrium-flutter, C_s2 Shock, Cardiaal
9. C_-VKF, atrium-flutter, C_Resp Insuff
10. C_Addisson / Bijnier Insuff, C_Oligurie (< 5 ml/kg/24u)
11. C_Addisson / Bijnier Insuff, C_s2 Shock, Cardiaal

Process Instances
40946
44830
45352
39578
45159
45161
45380
45267
44342
44723
45556
42984
41612
40430
42083
43303
40228
45402
39992
44475
44985
39953
45342
44659
41767
42807

Select FIS 3. C_Oligurie (< 5 ml/kg/24u)

Num of PIs satisfying this FIS: 12

Figure 6.5: Frequent Itemsets and their count as seen using the clustering option

6.3 Conclusion

The chapter discussed how clustering can be achieved in the ARM plug-in. Two illustrations were shown in the chapter. They showed that how clustering can be performed on the basis of association rules and frequent itemsets respectively. The first illustration showed that the obtained clusters can be used for further mining and the mined process models can be simple and easy to understand. Clusters can also represent frequent patterns existing in the log when the support count of items is chosen as the basis of clustering. Some more experiments were conducted using healthcare logs from Case study1, and it was observed that the process models from clustered PIs can be richer in dependencies thereby providing more insights into the underlying process. These clusters may be representative of the highly frequent behaviour found in the log or the exceptional medical cases in form of highly low frequent behaviour. Clustering can therefore be utilized to obtain specific process models or to generate simpler models as opposed to the spaghetti-like models. Further, the clustering functionality in ARM could also be enhanced to obtain a hierarchy of logs based on the association rules. The cluster from an association rule can be further mined with the ARM again, and one of the generated rules can be used for further clustering. Repeating this sequence of generating association rules and clustering can give the user a tree of logs satisfying different variants of the same association rule (that is initially used for clustering). The leaf nodes then would be the logs representing the most basic associations (may be in form of a rule: a=>b, instead of a=>b, c).

In the next chapter we introduce Case study2 and evaluate the performance of the ARM and its clustering mechanism on the data provided by this case study. The chapter will also give a comparison between the usage and performance of the HM, the DWS and the ARM for less structured healthcare processes.

7 Evaluation

Following the description and explanation of the ARM and experimental results in the previous chapters, this chapter aims at evaluating and analyzing the usefulness and applications of the ARM as it was implemented to obtain simpler process models for the healthcare domain as opposed to the complex models generated by the HM.

Case study2 is used in this chapter to perform testing of the ARM and to gain some insights in its processes by experiments with the ARM. In this chapter, we first describe this case study in Section 7.1 and illustrate the experiments with it in Section 7.2. In Section 7.3 the results of experiments with both of the case studies 1 and 2 are used to evaluate the ARM. Section 7.4 discusses the limitations of the ARM. The chapter is concluded in Section 7.5.

7.1 Case study2

As already mentioned in Chapter 1, Section 1.3.2, this case study consists of data from a preliminary study conducted on patients of acute stroke in 4 districts of Italian region of Lombardia, Italy[20]. The database contains records of patients suffering from the ischemic stroke problem. Information is recorded from the acute phase to the sub-acute phases of the patients suffering from the stroke. Acute phase data pertains to the data of patients that arrive at the hospital within 6 hours from the stroke symptoms onset. After the first 6 hours, the patient is considered to be in the sub-acute phase. The structure of the case study can be seen in the Entity-Relationship diagram given in Figure 7.1. The diagram presents following facts:

- Besides the patient's personal data being recorded, his clinical history is also recorded. It means, the complications that a patient suffered from in the past, treatments and measurements prescribed to him etc. are also kept in the records. Therapies received by the patients as a part of the previous treatments are also recorded.
- Whether a patient is admitted to the hospital while he is in acute phase or sub-acute phase, this information is recorded accordingly.
- All the measurements (laboratory tests etc.) and life parameters (like heart beat, blood gas etc.) are also recorded.
- All the treatment and therapies prescribed to a patient during his hospitalization phase are stored in the database.
- After the patient is discharged the post-treatment phase begins and data is then later on recorded for his follow-up visits to the hospital.

From the database it was also inferred that the number of events per case in Case study2 is smaller compared to the number of events per case in Case study1. This may be due to the fact that the Case study1 refers to the entire ICU where a patient suffering from numerous complications may be admitted, whereas the Case study2 refers only to the stroke patients, hence the number of events is less. Also the number of cases is very small i.e. 386. These cases are the patients suffering from stroke.

The data is stored in the MS-Access database. Therefore before it can be used for experiments with plug-ins in ProM, it was converted to the MXML format using the MS-Access import plug-in. Logs for experiments were made for 1) various therapies (medical, physical, acute phase and sub acute phase therapies) given to the patients and 2) various measurements done on patients for diagnosis and treatment purposes. Readers are referred to Appendix M to take a look at the important tables in this database. Fragments of logs used for experiments in this chapter can be found in Appendix N. The next section illustrates the experiments done on Case study2 to gain insights into the underlying healthcare processes related to stroke patients.

[20] We would like to thank Dr. Silvana Quaglini (Università degli Studi di Pavia, Italy), Dr. Anna Cavallini (IRCCS C. Mondino, Pavia, Italy), and Dr. Giusseppe Micieli (IRCCS Humanitas, Rozzano, Italy) for providing us with the Case study2 for experiments.

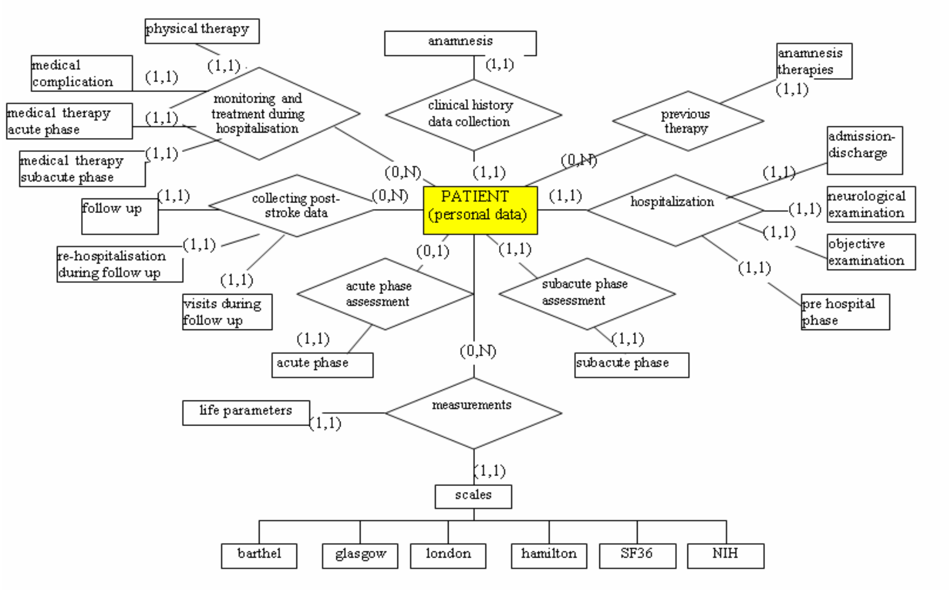


Figure 7.1: Structure of Case study2

7.2 Experimental results

7.2.1 Illustration 1

The log used in this experiment consists of information about various measurements (tests) done on stroke patients. This involves seven types of measurements viz., *Barthel, Glasgow, London, Hamilton anxiety, Hamilton Depression, SF36 and NIH* also indicated in Figure 7.1. The log has 373 PIs and 7 different ATEs. These PIs refer to patients that undergo these measurements. Each PI represents one single patient. The various ATEs in the log refer to the 7 types of measurements.

Before applying the ARM plug-in to this log, we first discovered the process model for the process registered in the log. Figure 7.2 gives the process model from the HM algorithm.

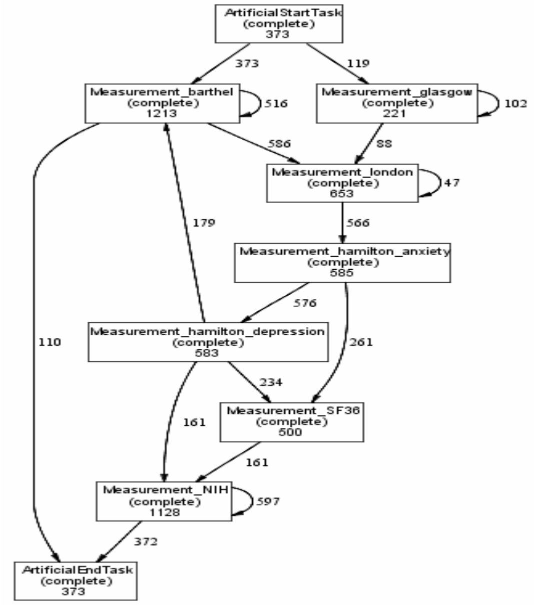
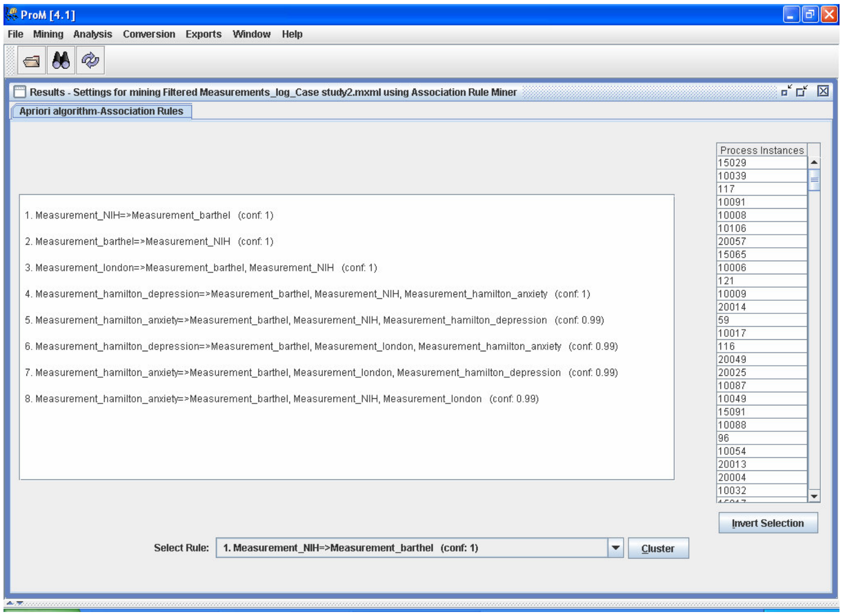

ArtificialStartTask (complete) 373
373
119
Measurement_barthel (complete) 1213
516
Measurement_glasgow (complete) 221
102
586
88
Measurement_london (complete) 653
47
179
566
Measurement_hamilton_anxiety (complete) 585
576
110
Measurement_hamilton_depression (complete) 583
261
234
161
Measurement_SF36 (complete) 500
161
Measurement_NIH (complete) 1128
597
372
ArtificialEndTask (complete) 373

Figure 7.2: Process model for measurements log

The process model is easy to understand and shows that each patient undergoes most of these measurements, may be on admission or during hospitalization. The patient is first measured for his extent of disability (*measurement Barthel*), and then his level of consciousness is recorded (*measurement Glasgow_coma_scale*). The model also shows close interrelationship between various measurements. For example, *measurement_Barthel* may be prescribed to a patient because of his level of depression recorded during *measurement_Hamilton_depression*. This may be due to the fact that depression may affect a patient's normal mobility to the extent that he is unable to eat, walk, bath etc. by himself.

Now the ARM is applied to this log in order to gain some more insights into this process of measurements. The results of applying the Apriori algorithm with a population size =100 can be seen in the Figure 7.3. We obtained 8 rules. These rules depict the implicit associations between various measurement activities in the log. For example, Rule 1 and 2 indicates that a patient undergoing M*easurement_barthel* always undergoes *Measurement_NIH*. Rule 3 indicates if a patient undergoes the test *Measurement_london*, he also undergoes M*easurement_barthel* and *Measurement_NIH*. The confidence of the rule is 1 indicating that all the patients that undergo the *Measurement_london*, also undergo the M*easurement_barthel* and *Measurement_NIH*. Such implicit information is not reflected in the process model in Figure 7.2. If this kind of information is available with the stakeholders i.e. the hospital staff then they can be well prepared with the skills and equipments needed to attend to such situations. This prior-information will help them to improve the quality of their medical services by saving their effort, time and cost of the services. Besides this information, some observations were also made from these association rules:

- The association rules also indicate a strong relationship between measurements like *Hamilton_depression*, *barthel, NIH* and *Hamilton_anxiety*. It should be noted that these rules do not involve measurements: *SF36* and *Glasgow_coma*. From the log statistics seen in ProM, it is found that the frequency of these measurements is the lowest. Therefore, when the association analysis is performed with lower value of support threshold (upper bound for minimum support is reduced from 1 to 0.8) rules involving these measurements are obtained.
- The presence of the activity *measurement_barthel* in all the generated association rules signifies its importance as compared to other measurements. This is also verified from the log summary. *Measurement_barthel* is the most frequent measurement followed by the *measurement_NIH*.
- For this process, the association rule algorithm captures all the events registered in the log. This may be an indication of the absence of noise or exceptional behavioural pattern in the log because it is quite possible that in case of noise/exceptional medical cases some events would not be captured.



Results - Settings for mining Filtered Measurements_log_Case study2.mxml using Association Rule Miner

Apriori algorithm-Association Rules

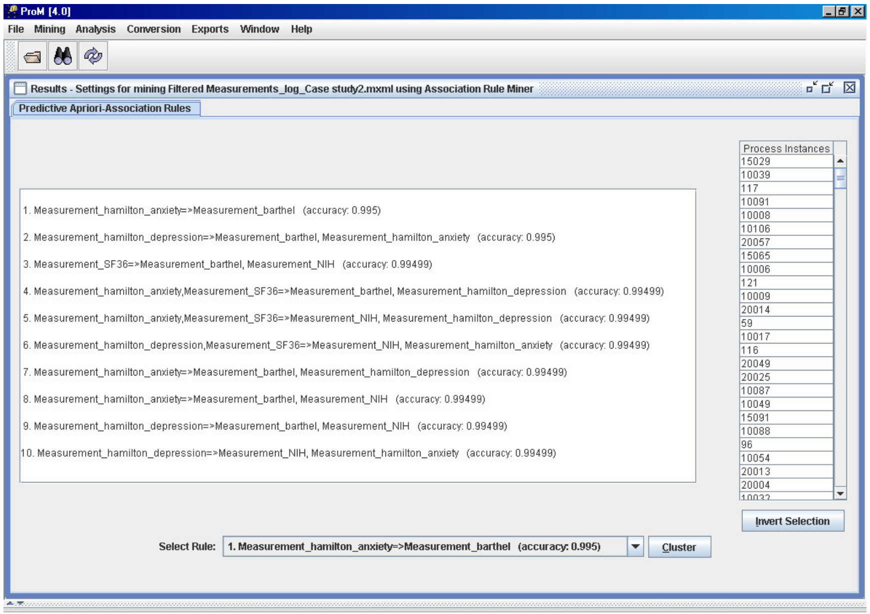
```
1. Measurement_NIH=>Measurement_barthel   (conf: 1)

2. Measurement_barthel=>Measurement_NIH   (conf: 1)

3. Measurement_london=>Measurement_barthel, Measurement_NIH   (conf: 1)

4. Measurement_hamilton_depression=>Measurement_barthel, Measurement_NIH, Measurement_hamilton_anxiety   (conf: 1)

5. Measurement_hamilton_anxiety=>Measurement_barthel, Measurement_NIH, Measurement_hamilton_depression   (conf: 0.99)

6. Measurement_hamilton_depression=>Measurement_barthel, Measurement_london, Measurement_hamilton_anxiety   (conf: 0.99)

7. Measurement_hamilton_anxiety=>Measurement_barthel, Measurement_london, Measurement_hamilton_depression   (conf: 0.99)

8. Measurement_hamilton_anxiety=>Measurement_barthel, Measurement_NIH, Measurement_london   (conf: 0.99)
```

Process Instances
15029
10039
117
10091
10008
10106
20057
15065
10006
121
10009
20014
59
10017
116
20049
20025
10087
10049
15091
10088
96
10054
20013
20004
10032

Select Rule: 1. Measurement_NIH=>Measurement_barthel   (conf: 1)

Figure 7.3: Association rules for various measurements prescribed to the stroke patients

Besides the Apriori algorithm, the PredictiveApriori algorithm was also used on this log. The rules obtained are shown in Figure 7.4. The predictive accuracy of the rules indicate their strength for unseen (not used for training) data too. As can be seen, the association rules obtained from the Apriori and the PredictiveApriori algorithms are different and also the metrics are different. In the Apriori algorithm the confidence values indicate the relative frequency of a correct prediction on the data that is used for training. Whereas, the PredictiveApriori uses the concept of predictive accuracy values obtained as a result of a trade-off between confidence and support to find an optimal way by maximizing the chance of correct predictions on unseen data.



Results - Settings for mining Filtered Measurements_log_Case study2.mxml using Association Rule Miner

1. Measurement_hamilton_anxiety=>Measurement_barthel (accuracy: 0.995)

2. Measurement_hamilton_depression=>Measurement_barthel, Measurement_hamilton_anxiety (accuracy: 0.995)

3. Measurement_SF36=>Measurement_barthel, Measurement_NIH (accuracy: 0.99499)

4. Measurement_hamilton_anxiety,Measurement_SF36=>Measurement_barthel, Measurement_hamilton_depression (accuracy: 0.99499)

5. Measurement_hamilton_anxiety,Measurement_SF36=>Measurement_NIH, Measurement_hamilton_depression (accuracy: 0.99499)

6. Measurement_hamilton_depression,Measurement_SF36=>Measurement_NIH, Measurement_hamilton_anxiety (accuracy: 0.99499)

7. Measurement_hamilton_anxiety=>Measurement_barthel, Measurement_hamilton_depression (accuracy: 0.99499)

8. Measurement_hamilton_anxiety=>Measurement_barthel, Measurement_NIH (accuracy: 0.99499)

9. Measurement_hamilton_depression=>Measurement_barthel, Measurement_NIH (accuracy: 0.99499)

10. Measurement_hamilton_depression=>Measurement_NIH, Measurement_hamilton_anxiety (accuracy: 0.99499)

Process Instances
15029
10039
117
10091
10008
10106
20057
15065
10006
121
10009
20014
59
10017
116
20049
20025
10087
10049
15091
10088
96
10054
20013
20004
10032

Select Rule: 1. Measurement_hamilton_anxiety=>Measurement_barthel (accuracy: 0.995)

Figure 7.4: Association rules using PredictiveApriori algorithm for measurements done on stroke patients

It is observed that the PredictiveApriori algorithm captures the low frequent activity: *measurement_SF36* in the first 10 rules it displays. However, it does not capture the lowest frequent activity: *measurement_Glasgow_coma* even when the number of rules is set quite high (30). This indicates a limitation of the algorithm because if a user gives less number of rules he would not be able to find rules involving the activity *measurement_Glasgow_coma*. It is quite difficult to know what number of rules should be set so as the low frequent activities or most of the activities registered in the log can be captured by the association rules.

In the next subsection, we describe our experiment with a log pertaining to various therapies given to the stroke patients.

7.2.2 Illustration 2

The log used in this experiment stores data about various therapies viz., physical therapy, surgical therapy, acute phase therapy and sub-acute phase therapy. It consists of 380 PIs and 35 different ATEs. Figure 7.5 shows a screenshot of the process model discovered by the HM algorithm.

It is apparent that the process model for the process of therapies is more complicated than the process of measurements. This may be because the number of events in the process of therapy is higher than the measurements process. Also, the various therapy events are very much interrelated. This may be a pointer to the complex process of treatment of stroke patients as a patient may need to be given multiple therapies at the same time or at different times during treatment. The process also consists of activities *medical_complications* indicating the complications which the stroke patients receiving various therapies suffer from. The presence of many length-one-loops indicates that it may be needed to repeat many therapies for certain the patients. This again indicates that the stroke treatment process is a complex procedure and requires long time.

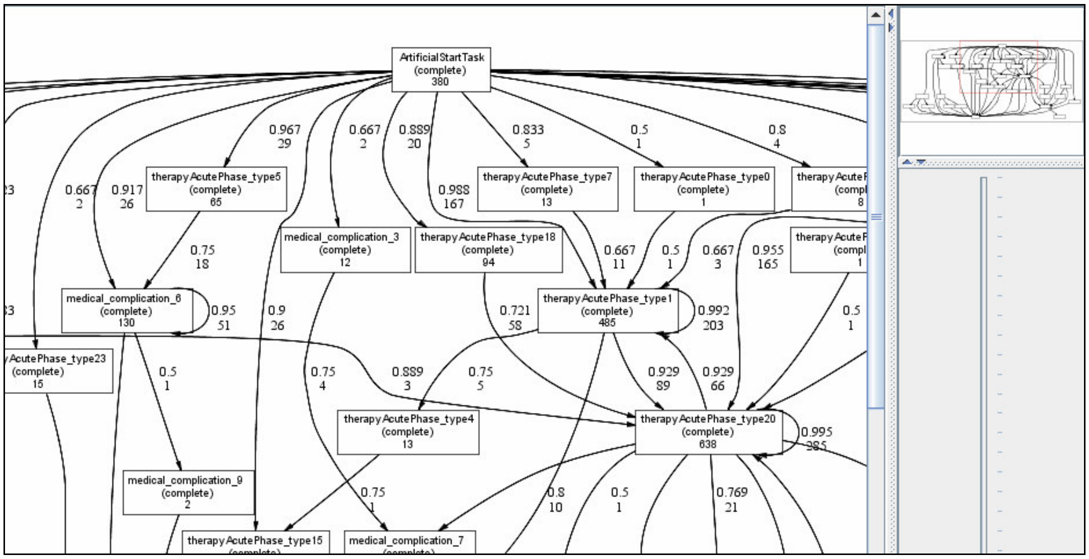

ArtificialStartTask (complete) 380
therapyAcutePhase_type5 (complete) 65
therapyAcutePhase_type7 (complete) 13
therapyAcutePhase_type0 (complete) 1
medical_complication_3 (complete) 12
therapyAcutePhase_type18 (complete) 94
medical_complication_6 (complete) 130
therapyAcutePhase_type1 (complete) 485
AcutePhase_type23 (complete) 15
therapyAcutePhase_type4 (complete) 13
therapyAcutePhase_type20 (complete) 638
medical_complication_9 (complete) 2
therapyAcutePhase_type15
medical_complication_7
0.967 29
0.667 2
0.889 20
0.833 5
0.5 1
0.8 4
0.667 2
0.917 26
0.988 167
0.75 18
0.667 11
0.5 1
0.667 3
0.955 165
0.95 51
0.9 26
0.721 58
0.992 203
0.5 1
0.5 1
0.75 4
0.889 3
0.75 5
0.929 89
0.929 66
0.995 385
0.75 1
0.8 10
0.5 1
0.769 21

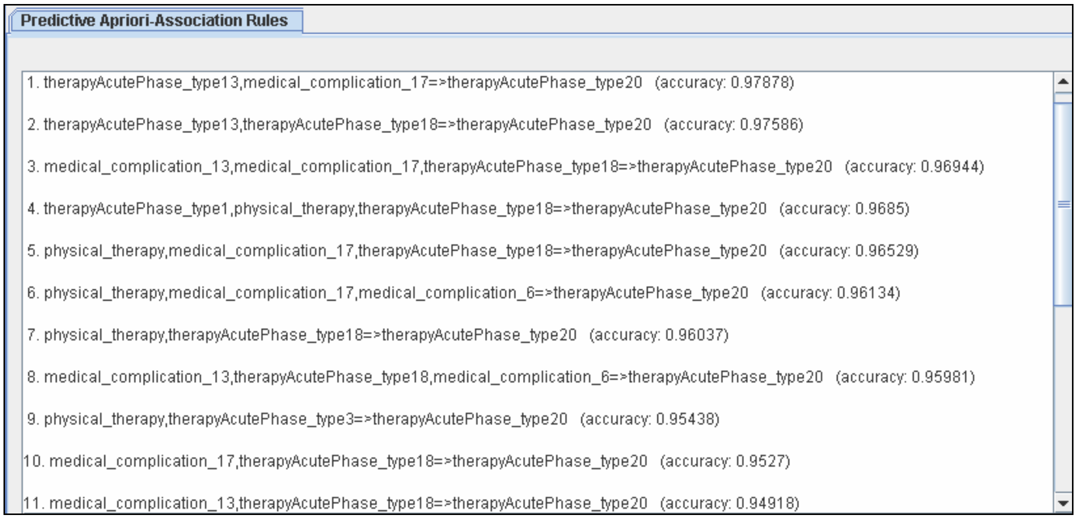
Figure 7.5: Process model for various therapies given to the stroke patients

Now we experiment with the ARM to see what insights can be gained for the complex stroke treatment process. The result of applying the ARM plug-in with a population size=50, confidence=0.5 we obtained 19 association rules with their confidence ranging from 0.95 to 0.73 (cf. Appendix O). Such a range of confidence values indicate that even rules with low confidence values indicate some correlation between the therapies/complications involved. For instance, consider the following association rules:

1. therapyAcutePhase_type20,therapyAcutePhase_type14=>physical_therapy (conf: 0.95)
2. therapyAcutePhase_type14=>therapyAcutePhase_type1 (conf: 0.88)
3. physical_therapy=>therapyAcutePhase_type20, medical_complication_13 (conf: 0.73)
4. therapyAcutePhase_type18=>therapyAcutePhase_type20 (conf: 0.73)

Rules 1 and 2 with confidence above 85% indicates that at least 85% of the times when a patient is given the therapies listed in LHS, he will also be given the therapies listed in the RHS. This represents the strong correlation between these therapies. Similarly, considering the Rules 3 and 4, it can be noted that 73% of the times when the patients receive therapies listed in the LHS, they also undergo therapies listed in the RHS. For these rules though the confidence value is lower than the confidence for the first two rules but nonetheless the latter rules also show a strong "implies" relationship between the various therapies they associate. It indicates that the therapies given to the patients are very much interrelated. This was also confirmed when the PredictiveApriori algorithm was applied to this log.

The PredictiveApriori algorithm also generated rules with a wide range of predictive accuracy values (cf. Figure 7.6) indicating a lot of correlation between various activities of the log. The rules from both the algorithms signify that the treatment process of stroke patients is a process involving various tasks (therapies) at the same time and a patient may be required to give many therapies in course of his treatment.



1. therapyAcutePhase_type13,medical_complication_17=>therapyAcutePhase_type20 (accuracy: 0.97878)

2. therapyAcutePhase_type13,therapyAcutePhase_type18=>therapyAcutePhase_type20 (accuracy: 0.97586)

3. medical_complication_13,medical_complication_17,therapyAcutePhase_type18=>therapyAcutePhase_type20 (accuracy: 0.96944)

4. therapyAcutePhase_type1,physical_therapy,therapyAcutePhase_type18=>therapyAcutePhase_type20 (accuracy: 0.9685)

5. physical_therapy,medical_complication_17,therapyAcutePhase_type18=>therapyAcutePhase_type20 (accuracy: 0.96529)

6. physical_therapy,medical_complication_17,medical_complication_6=>therapyAcutePhase_type20 (accuracy: 0.96134)

7. physical_therapy,therapyAcutePhase_type18=>therapyAcutePhase_type20 (accuracy: 0.96037)

8. medical_complication_13,therapyAcutePhase_type18,medical_complication_6=>therapyAcutePhase_type20 (accuracy: 0.95981)

9. physical_therapy,therapyAcutePhase_type3=>therapyAcutePhase_type20 (accuracy: 0.95438)

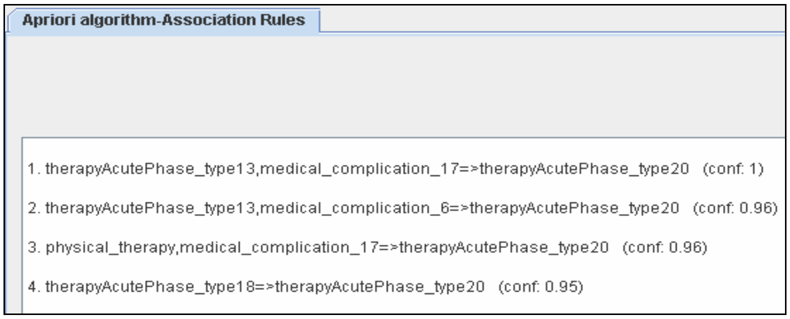
10. medical_complication_17,therapyAcutePhase_type18=>therapyAcutePhase_type20 (accuracy: 0.9527)

11. medical_complication_13,therapyAcutePhase_type18=>therapyAcutePhase_type20 (accuracy: 0.94918)

Figure 7.6: Wide range of predictive accuracy values obtained for the therapy process

7.2.3 Illustration 3

As already mentioned, the ARM also provides the functionality to cluster log traces. Clustering can help the user get smaller process models which represent either an association rule or a frequent itemset (as selected by the user). Below we show how clustering based on an association rule helped in obtaining a simpler process model. The log described in Section 7.2.2 is used to illustrate this. When the log is mined with the default parameter settings of the Apriori algorithm, 4 rules are obtained as shown in Figure 7.7.



1. therapyAcutePhase_type13,medical_complication_17=>therapyAcutePhase_type20 (conf: 1)

2. therapyAcutePhase_type13,medical_complication_6=>therapyAcutePhase_type20 (conf: 0.96)

3. physical_therapy,medical_complication_17=>therapyAcutePhase_type20 (conf: 0.96)

4. therapyAcutePhase_type18=>therapyAcutePhase_type20 (conf: 0.95)

Figure 7.7: Association rules for therapy log (default parameter settings)

Rule 3 is chosen to cluster the log in two parts: first part which contains all PIs satisfying this rule and the second part which do not satisfy this rule. Forty-seven PIs were found to satisfy this rule. We would like to show the differences in the process models before and after clustering. Figure 7.8 gives the process model of the entire log and Figure 7.9 gives the process model representing Rule 3:

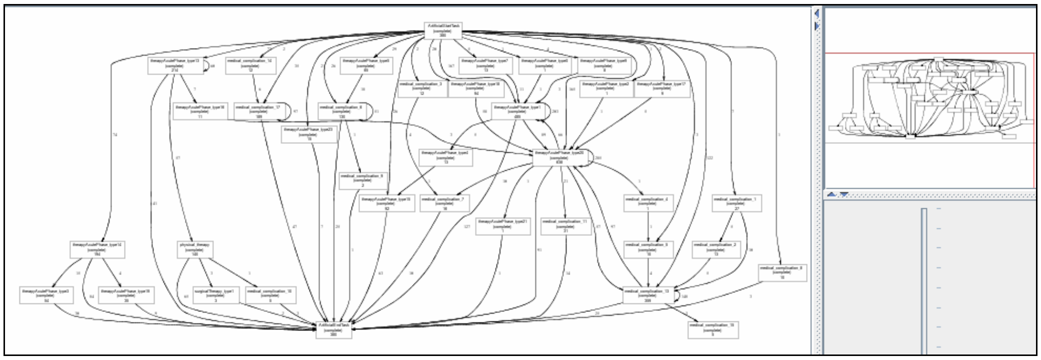


Figure 7.8: Process model for the complete therapy log

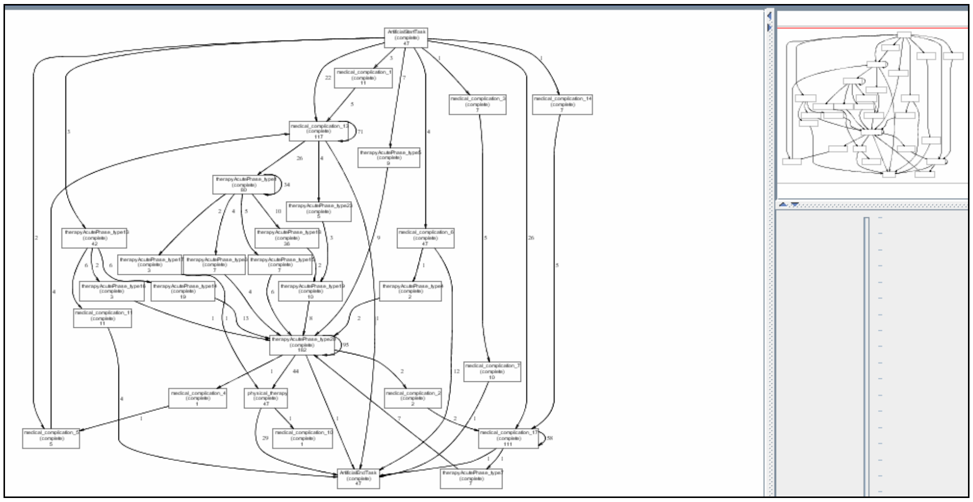
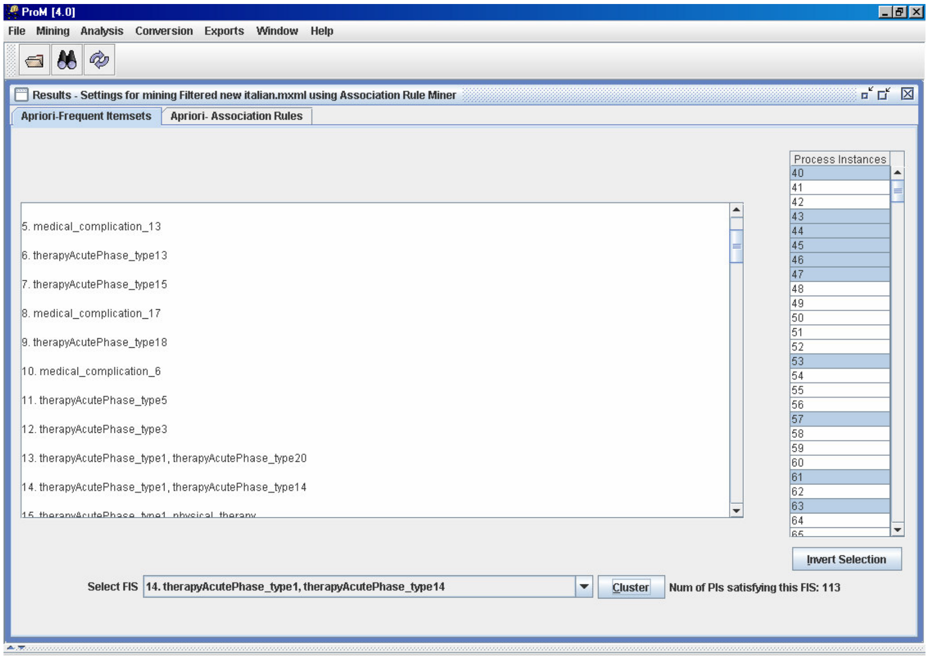


Figure 7.9: Process model specific for an association rule

It is apparent that the structure of the process in Figure 7.9 is simple as compared to the complex structure of the entire log in Figure 7.8. Therefore, simpler models obtained through clustering can be used for gaining insights into the process. Besides, as seen in Chapter 6, Illustration 2, clustering in the ARM also serves another purpose. It can also provide the support count of the antecedents and consequent itemsets in a rule. In Figure 7.9, it is seen that one of the frequent itemsets is the itemset-*therapyAcutePhase_type1, therapyAcutePhase_type14* and the number of process instances in which this itemset occurs (support count of this itemset, cf. Equation 4.2) in the event log can be obtained by selecting this itemset and clicking the button *Cluster*. This gives us the process instances which satisfy this itemset as well their count. The number of PIs satisfying this FIS is 113. Similarly, the support count for the task *medical_complication* is found to be 77.



Results - Settings for mining Filtered new italian.mxml using Association Rule Miner

5. medical_complication_13
6. therapyAcutePhase_type13
7. therapyAcutePhase_type15
8. medical_complication_17
9. therapyAcutePhase_type18
10. medical_complication_6
11. therapyAcutePhase_type5
12. therapyAcutePhase_type3
13. therapyAcutePhase_type1, therapyAcutePhase_type20
14. therapyAcutePhase_type1, therapyAcutePhase_type14
15. therapyAcutePhase_type1, physical_therapy

Process Instances
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

Select FIS 14. therapyAcutePhase_type1, therapyAcutePhase_type14

Num of PIs satisfying this FIS: 113

Figure 7.10: Frequent Itemsets and their count as seen using the clustering option

To summarize, process models for specific clusters show us a homogeneous group of patients who follow the care flow path represented by a particular association rule. When we obtained the process model for the entire log we used for this experiment, then it was seen that it is difficult to trace out a control flow path for the similar "characteristic" patients i.e. the patients undergoing the same care flow path in the process. When we find cluster describing an association rule and use it to mine a process model, the resulting process model is a specific and clean model which depicts homogeneity (cf. Figure 7.9). This is extremely useful in case of the healthcare domain because it is characterized by less-structured processes. These processes are also not unique as every patient represents a unique case and may or may not follow the same care path as followed by some other patient suffering from the same complication/taking up the same treatment or same test. Such heterogeneity of the cases makes it difficult to find one clear and understandable process model. This is where the ARM and the clustering technique find their importance.

After analyzing the ARM on Case study2 it is established that the association analysis has the potential to gain insights into less structured processes like healthcare. In the next sub-section the performance of the ARM is evaluated by establishing comparisons with the performance of the HM and the DWS.

7.3 The HeuristicsMiner & the DWS vs. the ARM

The limitations of the HM and the DWS were seen as the need to search for a process model representation that overcomes these limitations. Below we state these limitations and compare the results of the ARM with the results of these two algorithms.

The *HM* is one of the most robust algorithms available till date for logs containing noise and imbalance. Weijters et al. [28] conducted experiments with the HM on the benchmark artificial material. The result of these experiments emphasized the robustness of the algorithm in situations of noise and imbalance present in the log in various degrees. However the experiments were conducted on artificial material and using the default parameter settings of the algorithm. When the algorithm was applied to the two case studies we found that the results of the HM are not what they were expected to be. The purpose of the algorithm is to discover the process model underlying the investigated process. We performed many experiments with different parameter settings but the algorithm failed to provide a clear and understandable process model. The process models obtained were complex and full of problems like missing activities (activities registered in the event log but not captured in the process model), missing dependencies and dangling activities (though the artificial start and end tasks were added to the logs). When the healthcare logs were mined without using the *all-activities connected heuristic* the process model obtained for some logs were better in terms of simplicity. But these models were full of disconnected and dangling activities, and therefore these models do not exhibit those connections that were shown by the models generated using the *all-activities connected heuristic* parameter. For some logs, only a list of disconnected activities is provided by the algorithm. So, it could not be concluded whether not using the *all-activities connected heuristic* is a good choice. Besides, unclear joins/splits i.e. mixed AND/XOR were also observed in case of AND/XOR semantics. Based on all these observations we concluded that the dependency graph is not an appropriate model representation for less structured data from the healthcare domain. These limitations of the HM in context of healthcare processes led us to exploration of another existing algorithm in the ProM framework, the DWS plug-in. The DWS plug-in not only gives a process model but also provides knowledge about the underlying process in terms of some behavioural patterns contained in the log.

The *DWS* approach discovers a set of workflow models that represents different subsets of the input log. The mining is carried out through a top-down hierarchical clustering process, where the log is recursively split into homogeneous clusters (from a behavioural viewpoint). All discovered clusters are then equipped with a specific workflow model using the HM. Any partitioning step hence produces a refinement of the workflow model being discovered. Specific behavioural patterns, named discriminant rules, are used as features for clustering log instances by means of classical k-means algorithm [33]. These discriminant rules represent global constraints in the event log. An event log from the healthcare domain when used with the HM may be full of problems mentioned above. But an event log mined with the DWS plug-in delivers some behavioural patterns which are not noticeable from the dependency graph generated by the HM. These patterns provide insights into the process model but they have limitations of their own. These rules can not handle loops and only involve adjacent tasks. The rules are based only on the relationship of the neighbouring tasks which means the relationships between the non-adjacent tasks are ignored. Moreover, given the frequency parameters sigma and gamma, the rules are not easy to comprehend. So, the exploration of the DWS plug-in also did not provide us knowledge that could be readily used by the stakeholders for improving their services in terms of cost, time and effort. This motivated us to look for yet another process model representation so that we could get some meaningful information out of the healthcare data. Therefore we experimented with the Case Data Extraction plug-in implemented in the ProM in combination with the Weka machine learning library and proposed a new mining plug-in that could generate association rules, a very popular classical data mining technique.

Association rules in context of process mining give associations between tasks in an event log on the basis of parameters like support and confidence. In chapters 5, 6 and 7 we experimented with the *ARM* plug-in and analyzed the association rules obtained from it. We saw that the ARM plug-in provide insights into the underlying process in form of association rules and these rules generally go unnoticed as this information is beyond the pure execution of the activities/tasks in the log and is un-captured by most of the mining plug-ins.

When we compare the DWS and the ARM plug-ins, it is found that the output of both plug-ins are the frequent patterns found in an event log. But the DWS approach misses out the relationship between the non-adjacent tasks whereas the ARM gives us the association rules involving both adjacent and non-adjacent tasks. Moreover, the DWS parameters: sigma and gamma make it hard to comprehend the discriminant rules easily. But the association rules simply state the fact that if a task executes, what other tasks would also execute or have been executed. The ARM also scores over the DWS approach because: first, although the DWS approach generates clusters and the discriminant rules contained in them, information like what percentage of traces in the cluster satisfies which rule is missing. Second, the association rules from the Apriori algorithm are ranked based on their confidence values. So, a rule with higher value of confidence is strong and more reliable than a rule with lower confidence. The association rules generated by the PredictiveApriori algorithm are also ranked based on the predictive accuracy. The DWS plug-in though provides the rules inn order of their importance, but the information about their relative importance is not quantified. Third, the ARM also offers the feature of clustering the event log based on a particular association rule/frequent itemset. These clusters can be further used for other mining algorithms. Process models specific to these clusters can be generated using mining plug-ins like the HM, the Alpha algorithm etc. available in ProM, and these clusters can also be analyzed using the analysis plug-ins. The organizational perspective of these clusters can also be mined using the Social network miner and similar plug-ins. So, we see that the association rules combined with the feature of clustering can generate an event logs which can be reused or further used for several mining and analysis algorithms (including the ARM itself). Whereas, the different subsets of the input log generated by the DWS plug-in cannot be reused for mining. Only analysis plug-ins available in ProM can be applied to these subsets process models.

The ARM however can’t be compared to the results of the HM as they both generate different process models. The HM generates a dependency graph and the ARM generates association rules. The limitations of the HM therefore are not dealt with in the ARM. The ARM just provides an alternate process model representation different from the one based on the pure control flow. The ARM gives insights into the process but not in form of a visual process model like the Petri nets or the dependency graph. However, these process models can be obtained using the clustered PIs from the ARM using some mining plug-in available in ProM. Our purpose behind the proposal and implementation of the ARM was not to replace the HM but was to obtain behavioural insights into the underlying process. This behaviour is not explicitly presented in a dependency graph. Moreover, it should be noted that the ARM is able to deal with noise if noise refers to the errors done in recording the activities in their proper execution order. As the association rules only represent the associations between the activities purely based on their execution, therefore if any log does not have the ‘timestamp’ information, the association rules would still be consistent. This was observed in the Case study2 because in the case study, the ‘timestamp’ information of the activities is missing and only the date is recorded. If such logs are given to the HM and the DWS algorithms, they may not be able to generate correct models as the dependency graph will still depict the erroneous dependencies, and also the discriminant rules will portray wrong results.

Though the ARM seems to be a good approach for mining flexible processes, it can be improved to provide better results. The next section outlines the current limitations of the ARM.

7.4 Limitations of the ARM

The limitations of the ARM relates to the limitations of the algorithms available in it. These limitations are stated below:

- Every association rule algorithm first generates frequent itemsets and then derives association rules from these frequent itemsets. Computational requirements for frequent itemset generation are generally more expensive than those of rule generation. When the value of the support thresholds is lowered it results in more itemsets declared as frequent. This increases the computational complexity of the algorithm because candidate itemsets must be generated and counted. The maximum size of frequent itemsets also increases with lowering the support threshold values. With this increase, the algorithm has to make more passes over the dataset. The total number of iterations required by the algorithm is k_{max} +1, where k_{max} is the maximum size of the frequent itemsets. Therefore, when lower support values are given to the algorithm to generate rules involving low frequent activities, the computational complexity also increases, thereby degrading the performance of the ARM in terms of computation time.
- When more itemsets (activities or group of activities) are declared as frequent itemsets more space is needed to store the support count of these items. This increase in the number of events increases the computation and I/O costs as larger number of candidate itemsets will be generated by the algorithm (cf. Section 5.3, Figures: 5.14, 5.15).
- The Apriori algorithm makes repeated passes over the data set therefore its run time increases with the size of the dataset. If the number of PIs in the event log is large the runtime for the algorithm also increases. For the PredictiveApriori algorithm also the computation time increases linearly with an increase in the number of PIs (cf. Section 5.3, Figure 5.12).
- Also if the width of PIs is large i.e. the number of ATEs contained in a PI is large then the number of hash tree traversals performed during the support counting is also increased. (In the Apriori algorithm candidate itemsets are partitioned into different buckets and stored in a hash tree). This also consumes a lot to time.
- The association rule algorithms do not deal with length-one-loops. For example, if an event log contains traces of the type "…aa…" it does not generate a rule showing that the task 'a' is in loop with itself. So, this information is missing in the ARM. But it is capable of dealing with loops involving more than 1 task i.e. length two or three loops.
- The confidence measure ignores the support of the itemset in the rule consequent. Due to this some high confidence rules can sometimes be misleading [25]. A better metric like the *lift* can be used to indicate interesting rules. Lift is a metric that also considers the support count of the RHS items in an association rule.
- Practically, by varying the values of the confidence and support parameters in the Apriori algorithm hundreds of association rules can be generated. But many of these rules are redundant and do not provide any new information. So, the search of interesting and non-redundant association rules is a very popular research topic. In the ARM though we use the original Apriori algorithm but we apply our concept of interesting rules (stated in Section 5.2.3) to retain only the non-redundant rules. That means we still take the output of the Apriori and then apply our filters. This consumes extra memory and time as many frequent itemsets are computed without any use as the rules that may be generated from them are eventually discarded because they may be redundant rules. It means a new concept for the association rule algorithm should be proposed that generates only the non-redundant rules and the frequent itemsets are generated accordingly.
- The frequent itemset generation in the PredictiveApriori algorithm is also computationally expensive. It can be improved using the approach used in the AprioriTid algorithm (cf. Section 4.1.2). The latter minimizes the number of database passes by representing the transactional dataset in vertical layout (storing the list of transaction identifiers) rather than the horizontal layout (storing the transactions themselves.

7.5 Conclusion

In this chapter we presented our experiments with the Case study 2 using the ARM plug-in. We analyzed the association rules and understood the strengths of these rules indicated by confidence and predictive accuracy metric. It was established that association rules and frequent itemsets represent behavioural and frequent patterns in event logs which are not explicitly communicated by a process model (like Petri net, dependency graph etc.) mined using some mining algorithm. These frequent patterns can be further used as a criterion for clustering the event log into clusters that satisfy a particular behavioural pattern and the ones that do not satisfy this pattern. These clustered PIs can be supplied as input to some mining algorithms to gain specific process models exhibiting homogeneity of cases. We also compared the HM and the DWS algorithms with the ARM and, analyzed the importance and usefulness of the ARM with respect to these algorithms. In spite of the limitations of the association rule algorithms implemented in the ARM, the ARM can be used for mining complex and less-structured processes from domains like the healthcare to determine homogeneous care flow paths. In the next chapter, we conclude the work done in this graduation assignment by stating the contributions and the future work that follows this research project.

8 Conclusion

This chapter concludes the research work done in this graduation assignment. Section 8.1 summarizes the findings from the various experiments done in this thesis with the newly implemented association rule mining plug-in i.e. the ARM developed for ProM. Section 8.2 states the insights into the medical domain that we gained from the experiments with the ARM. In Section 8.3 we mention the contributions made through this research assignment and Section 8.4 discusses the future work following this research work.

8.1 Summary

In this section, we reflect on the work done to achieve the research goals stated at the beginning of this thesis.

Our first and second research goals targeted at evaluating the performance of the process mining algorithms on the healthcare processes. Towards this we evaluated the mining algorithms: the HM and the DWS on data from two healthcare organizations: the Catharina hospital and the data from study of Stroke patients in the Italian region of Lombardia. We concluded that though the HM algorithm is one of the most robust algorithms, it is inappropriate for healthcare domain. This is so because the processes in healthcare are less-structured (less regularity of control flow paths), cross-functional and multi-disciplinary in nature and when the HM is applied to such processes the result id 'spaghetti'. The complex process models with dangling, disconnected and missing activities, and missing dependencies makes it difficult for the user/stakeholders to extract any information from them. Also, the HM cannot deal with mixed AND/XOR situation which is typical of healthcare domain. All these factors point out that HM is not suitable for mining flexible processes. After experimenting with the HM, discriminant rules were viewed as an alternate process model representation. The DWS plug-in uses the HM algorithm to generate process models. Therefore, though we have additional insights about the process model in terms of the discriminant rules the process models generated by the HM for these rules have the same problems we mentioned above. Moreover, these rules are difficult to comprehend and do not take into account the non-neighbouring tasks.

Our third research goal stems from these problems, and focuses on investigating the classical data mining technique of association rules. For this a mining plug-in, the Association Rule Miner has been implemented in the ProM framework. This plug-in not only gives us insights into the process underlying an event log through the association rules but also gives the option to cluster the event log based on the criteria of association rules and frequent itemsets. The ARM plug-in utilizes the association analysis algorithms: the Apriori and PredictiveApriori implemented in the Weka machine learning library. However, to the rules derived from the Apriori algorithm we applied our own approach of retaining non-redundant rules. The functionality of clustering caters to our fourth research goal. Clustering is viewed as a mechanism to retrieve homogeneous group of PIs where these PIs belong to the same control flow path. Using the combination of association rules and clustering we can retrieve group of patients following the same care flow path for some complications, treatment or measurements. This helped us to partition an event log into parts, each part revealing a distinct structure and meaningful information about the healthcare process. Moreover, these clusters can be put to different use by giving them as input to various mining and analysis algorithms to throw more light on the less structured processes in the healthcare domain. Also because association rules are not based on the execution order of the activities, the ARM can deal with errors in recording the activities in wrong execution order because the association rules indicate only the presence of an activity if some other activity (-ies) is also present. Given all the above factors, we find the use of the ARM quite promising for flexible processes. In the next section we present the insights from the medical domain obtained during this research assignment.

8.2 Insights from medical domain

In this section we present the insights gained in this research assignment. We present the information gained and interpreted while using the two case studies for various mining algorithms.

- We observed that Case study1 has a lot of low frequent events. This is because it contains data from the ICU where severely ill patients or the patients with multiple diseases are admitted. For such patients, the treatment process cannot be standardized or known beforehand. Also, the complication paths followed by these patients are not known in advance. This uniqueness of the patients is one of the main causes of complex models for this case study.

- The process models obtained for the Case study2 are less complex as compared to models for Case study1. This may be because: 1) the size of Case study2 is much smaller than the size of case Study 1, 2) number of events in Case study1 is much larger than the number of events in Case study2, and 3) Case study2 contains data pertaining only to the stroke patients whereas, the Case study1 contains data about numerous complications that the patients admitted to ICU may suffer from (unknown or flexible complication paths).
- For Case study1:
  - Association rules indicating complications that always occur together were obtained. For example, from Section 5.2.1, Illustration 1 it was interpreted that the complications *C_-VKF, atrium-flutter* and *C_Oligurie (<5ml/kg/24u)* always occur together.
  - It was also found that patients receiving treatment *B_Thoraxdine* always received the treatment *B_Beademing* also.
- For Case study2:
  - It was discovered that the process of measurements is much simpler than the process of therapies.
  - Every stroke patient undergoes the treatment *Measurement_Barthel* as indicated from Section 7.2.1, Illustration 1, Figure 7.3.
  - Process model in Figure 7.5 and association rules in Section 7.2.2, Illustration 2 concludes that the various therapies given to the stroke patients are very interrelated to one another and a patient is given multiple therapies during his course of treatment.

8.3 Contributions

In the previous sections, we already concluded our observations and findings from the experiments done on the HM, the DWS algorithm, and the ARM, and the medical insights obtained from them. This section explicitly outlines the contribution made to the ProM research group.

- The HM is the one of the most robust mining algorithms in the ProM research area. We applied the HM on real-life logs in form of the healthcare data and discovered that it is not suitable to mine flexible and less structured processes. It is also discovered that the HM is unable to deal with unclear AND/XORs. In the case of unclear AND/XOR the fitness measure indicating the quality of a process model loses its importance because the model itself is not correct. We found that the dependency graph is inappropriate to mine process models for less structured processes.
- We combined the process mining research area with the data mining domain by incorporating the classical data mining technique of association rules in the Process Mining framework.
- We proposed to apply the concept of association rules to the healthcare domain to gain insights into the dynamic and flexible healthcare processes and, in combination with clustering we enabled the partitioning of an event log into homogeneous groups of patients.
- We provided two association analysis algorithms in ProM: the Apriori and the PredictiveApriori. The Apriori algorithm is the simplest algorithm to date and was the first one to use support-based pruning to systematically control the exponential growth of candidate itemsets [25]. The PredictiveApriori algorithm was provided because sometimes it is difficult to specify values for different thresholds like support, confidence etc. (in threshold-based algorithms like the Apriori) and this algorithm asks for the most natural parameter from the user i.e. the number of rules the user wants to generate from the event log.
- As already mentioned in Section 8.1, the evaluation of the ARM on Case studies 1 and 2 presents insights into the medical domain. It was established that these case studies comply with the characteristics of the healthcare domain explained in Section 1.2. Both the case studies reveal the flexible and dynamic nature of the domain. They also represent the uniqueness and heterogeneity of the cases. It was also made explicit that the healthcare organizations have a lot of events but most of these events are highly low frequent.

- The development of the ARM plug-in serves as a starting point for further research and experimentation with domains characterized by less structured processes.

8.4 Future Work

After presenting the contributions of this thesis, we present some possible directions of future research:

- We observed that the process models discovered by the HM contained dangling activities, unclear AND/XOR join/splits and missed activities and dependencies. Additional research is required to insure the discovery of process models which are free from these problems.
- Currently, clustering is done on the basis of single frequent itemset/association rule. In future it can be done on the basis of multiple itemsets and rules. This way we will be able to capture the overlapping process instances too. Also, the concept of hierarchical clustering based on association rules can be implemented.
- Sequential pattern mining as proposed by R. Agrawal and R. Srikant can also be implemented in the ARM. Sequential pattern mining aims at finding frequent sequences of itemsets in a dataset. In context of process mining, this can be utilized to find frequent sequences of events; this will provide the user with the events that always happen together and with the same frequency.
- By modifying the algorithms implemented in the ARM to be memory efficient, they can be used for datasets with more width (i.e. more number of ATEs per PI) and larger number of PIs. The ARM can be used not only for healthcare processes but also for Web Mining (a dynamic domain) where the user behaviour may vary significantly across time and old access patterns may be no longer relevant. Also, the association rule algorithms implemented in the ARM can be extended to incremental association rule mining where the previously mined information can be re-used and combined with the fresh data to efficiently re-compute the new set of association rules.
- The ARM presents to the user the Boolean association rules that are based on the presence/absence of an ATE in a PI. This is one of the criteria that can be used to group similar patients. This can be extended to compute association rules on the basis of data attributes. For example, an association rule which implies that a patient of age 60 suffering from obesity will also suffer from diabetes, can be retrieved. This association rule emphasizes on the age of the patient. Such rules along with clustering will enable us to get a group of patients on the basis of their personal attributes besides the medical attributes (complications, treatments etc.).
- Other future directions include using different metrics besides the support and confidence. Lift, interest factor etc. can be used to retain interesting rules.

Epilogue

In this epilogue, I take the opportunity to express my personal view on this research assignment and the results obtained from it.

Firstly, I found this research assignment very challenging and it was interesting every day for the last one year. In the beginning, it was difficult to find the way forward. The language of the Case Study 1 being Dutch and the medical 'jargon', were the initial problems that I faced. With time after overcoming these problems, spaghetti-models from the HM algorithm posed another problem. As the main objective of this research assignment was to gain insight into the healthcare processes, such complex models were seen as obstacles. But, this complexity finally led me to the core topic of this research and I proposed to use more traditional data mining techniques within the process mining domain.

My proposal to use Association Rules to gain insights into the underlying processes encouraged me to move ahead with enthusiasm and confidence as this reflected the innovative and creative side of the research assignment. But the greatest struggle in the project started from here: the implementation of the proposed idea in form of the Association Rule Miner plug-in. The lack of programming skills in Java was my biggest handicap and therefore the implementation took much more time than expected. At the beginning of the implementation, I found it difficult to organize/structure the plug-in (in terms of what classes to design, how they will interact with one another etc.). I was also unfamiliar with the ProM classes and their usage. Interfacing with Weka library was also a challenge as I had to integrate in-built classes from ProM, Weka and the classes which I defined. All these factors resulted into less available time for interpreting the results and performing their actual analysis. It would have been nice if there had been more time for this, so that the importance of the plug-in could be explored further. In this case, the authorities at the two healthcare organizations (case studies) would have been contacted to discuss the findings. However, I feel that a nice approach (association rules) has been implemented in the ProM framework and it opens up numerous possibilities for mining less structured processes.

I also feel that this report provides an easy to understand introduction to the domain of association rules. The reader gets a clear view of the importance of the association rules in context of the problem definition given in Chapter 1, Section 1.2. I would also like to mention that problems and bug within the HM algorithm has been brought into light especially its inability to deal with the less structured processes. Some bugs within the Case Data Extraction mining plug-in have also been reported to the author of this plug-in. Besides, I am very glad to succeed in my implementation (using Java) effort and this has been learning for life.

As far as the results of mining the data from the two case studies using the Association Rule Miner are concerned, a lot of improvements can follow. I would like to say that in context of a typical research assignment it is difficult to obtain desired results in the end. The outcome of any research can be conclusive or it lays foundation for future exploration. But the important thing is that a new concept has been proposed, motivated and implemented. I also tried to improve the association rules generated from the Apriori algorithm by applying some filtering criteria (Section 5.1.2). This resulted into providing the user with non-redundant association rules. But it seems, this criteria needs to be further improved to get interesting rules besides non-redundant rules. All the proposed improvements point to the potential of the association rules to create a better understanding of the flexible, dynamic and less-structured processes not only from the healthcare domain but any other domain with such processes.

In the end, I must say that for me this research assignment was quite difficult, challenging and interesting all at the same time. I am very glad to learn so many things, both tangible and intangible. I learnt the concepts of process mining and data mining, and learnt by experience how to integrate with third party software (Weka library in my case). I also indirectly learnt project management skills from the mistakes I made during this one year. I also gained working knowledge of the Java programming. Finally, I would like to mention, last but not least, that the research environment at the IS department motivated and inspired me every time to not give up and keep moving ahead with confidence and my goal in focus.

Bibliography

[1] W.M.P. van der Aalst, A.H.M. ter Hofstede, M. Weske. *Business Process Management: A Survey*. In: Proceedings of the 2003 International Conference on Business Process Management (BPM2003). Lecture Notes of Computer Science 2678. Berlin, Springer-Verlag, pp.1-12, 2003.

[2] W.M.P. van der Aalst. *Making Work Flow: On the Application of Petri nets to Business Process Management*. In J. Esparza and C. Lakos, editors, Application and Theory of Petri Nets 2002, volume 2360 of Lecture Notes in Computer Science, pages 1-22. Springer-Verlag, Berlin, 2002.

[3] 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.

[4] W.M.P. van der Aalst. *Three good reasons for using a Petri-net-based workflow management system*. In S. Navathe and T. Wakayama, editors, Proceedings of the International Working Conference on Information and Process Integration in Enterprises, pages 179--201, Cambridge, Massachusetts, 1996.

[5] W.M.P. van der Aalst and A.J.M.M. Weijters. *Process mining: a research agenda.* In Computers in Industry 53, Elsevier B.V., 2003.

[6] W.M.P. van der Aalst. *Business Process Management: A Personal View.* Business Process Management Journal, 10(2):135-139, 2004.

[7] R. Agrawal and R. Srikant. *Fast Algorithms for Mining Association Rules*. In Proc. of the 20th Int'l Conference on Very Large Databases, Santiago, Chile, September 1994.

[8] M. J. Berry, and G. Linoff. *Data Mining Techniques: for Marketing, Sales, and Customer Support*. John Wiley & Sons, Inc, 1997.

[9] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. 1997. *Dynamic itemset counting and implication rules for market basket data*. In Proceedings of the 1997 ACM SIGMOD international Conference on Management of Data (Tucson, Arizona, United States, May 11 - 15, 1997). J. M. Peckman, S. Ram, and M. Franklin, Eds. SIGMOD '97. ACM Press, New York, NY, 255-264.

[10] B. F. van Dongen, A.K. Alves de Medeiros, H.M.W. Verbeek, A. J. M. M. Weijters, and W. M.P. van der Aalst. The prom framework: A new era in process mining tool support. In *ICATPN*, pages 444-454, 2005.

[11] M. Dumas, W. M. P. van der Aalst, and A. H. ter Hofstede. *Process-Aware Information Systems: Bridging People and Software Through Process Technology*. John Wiley & Sons, Inc. 2005.

[12] G. Greco, A. Guzzo, and L. Pontieri. *Discovering Expressive Process Models by Clustering Log Traces*. IEEE Transactions on Knowledge and Data Engineering 18, 8 (Aug. 2006), 1010-1027.

[13] C. W. Günther and W.M.P. van der Aalst. *Process Mining in Case Handling Systems*. Paper presented at the Proceedings of the Multikonferenz Wirtschaftsinformatik 2006 (MKWI 2006), Passau, Germany.

[14] N. Jayaratna. *Understanding and Evaluating Methodologies: Nimsad-A Systemic Framework*. McGraw-Hill, UK, 1994.

[15] R. Lenz and M. U. Reichert. *IT Support for Healthcare Processes.* In: Proceedings 3rd International Conference on Business Process Management (BPM'05), 5 - 8 Sep 2006, Nancy, France. pp. 354-363. Lecture Notes in Computer Science 3649. Springer Verlag.

[16] A.K. Alves de Medeiros. *Genetic Process Mining*. PhD Thesis. Technische Universiteit Eindhoven, Eindhoven, The Netherlands.

[17] A. K. Alves. de Medeiros, B. F. van Dongen, Wil M. P. van der Aalst, A. J. M. M. Weijters. *Process Mining for Ubiquitous Mobile Systems: An Overview and a Concrete Algorithm*. UMICS 2004: 151-165

[18] G. Micieli, A. Cavallini, S. Quaglini. *Guideline Complicance Improves Stroke Outcome, A Preliminary Study in 4 Districts in the Italian Region of Lombardia*. Stroke 2002;33:1341-7.

[19] B. Mirkin. *Clustering for data mining: a data recovery approach*. Publisher London: Chapman and Hall/CRC, 2005.

[20] T. M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997

[21] M. Poulymenopoulou, F. Malamateniou and G. Vassilacopoulos. *Specifying Workflow Process Requirements for an Emergency Medical Service.* J. Med. Syst. 27, 4 (Aug. 2003), 325-335.

[22] A. Rozinat and W.M.P.van der Aalst. *Decision mining in ProM*. In S. Dustdar, J.L. Fiadeiro, A. Sheth , Business Process Management (Proceedings 4th International Conference, BPM 2006, Vienna, Austria, September 5-7, 2006) (Lecture Notes in Computer Science, Vol. 4102, pp. 420-425). Berlin: Springer.

[23] A. Rozinat and W. M. P. van der Aalst. *Conformance Testing: Measuring the Fit and Appropriateness of Event Logs and Process Models*. Business Process Management Workshops, 2005, 163-176.

[24] T. Scheffer. *Finding Association Rules That Trade Support Optimally against Confidenc*e. *In Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery* (September 03 - 05, 2001). L. D. Raedt and A. Siebes, Eds. Lecture Notes In Computer Science, vol. 2168. Springer-Verlag, London, 424-435.

[25]P. Tan, M. Steinbach and V. Kumar. *Introduction to Data Mining*. Addison Wesley, 2006.

[26] H.M.W. Verbeek, B.F. van Dongen, J. Mendling, W.M.P. van der Aalst. *Interoperability in the ProM Framework*. In: CAiSE 2006 Workshop Proceedings - Open INTEROP Workshop on Enterprise Modelling and Ontologies for Interoperability (EMOI-INTEROP 2006), 619-630, June, 2006

[27] V. Weerakkody and W. Currie. *Integrating Business Process Reengineering with Information Systems Development: Issues & Implications.* In: Proceedings of Business Process Management Conference (BPM), Eindhoven, Netherlands, June 2003.

[28] A.J.M.M. Weijters, W.M.P. van der Aalst, and A. K. Alves de Medeiros. *Process Mining with the HeuristicsMiner Algorithm*. BETA Working Paper Series, WP 166, Eindhoven University of Technology, Eindhoven, 2006.

[29] T. Wendler, and C. Loef, *Workflow management-integration technology for efficient radiology*. Medicamundi 45(4):41-48, 2001.

[30] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[31] http://www.albionreserach.com/data_mining/market_basket.php

[32] A Tutorial on Clustering Algorithms.
http://www.elet.polimi.it/upload/matteucc/Clustering/tutorial_html/index.html

[33] http://en.wikipedia.org/wiki/Business_activity_monitoring

[34] The UbiWFMS Project: Workflow Management and Process Automation in Pervasive/Ubiquitous Environments. www.ifi.unizh.ch/dbtg/**Projects**/**UbiWFMs**/index.html

[35] http://www.informatik.uni-freiburg.de/~ml/IDB/talks/Sacca_slides.ppt.

[36] http://is.tm.tue.nl/research/processmining/WorkflowLog.xsd

[37] The Disjunctive Workflow Schema plug-in reference in the ProM framework. www.processmining.org

[38] User Manual for converting data from MS Access database to the ProM MXML format. http://promimport.proessmining.org

[39] Perspectives on Information Technology for the Health Care Industry. Tunitas group, Workflow Automation. http://www.tunitas.com/pages/Workflow/workflow.htm

[40] Workflow Management Coalition (The Workflow Reference Model) http://www.wfmc.org/standards/docs/tc003v11.pdf

[41] http://www.win.tue.nl/~hverbeek/prom-plugins.html#mining

[42] Introduction to Workflow in Healthcare. (Healthbase Australia) http://workflow.healthbase.info/wf_in_healthcare.html

A: Process mining framework and MXML format

To make different mining tools to have a common input format, the XML tool independent format was introduced. Examples of different mining tools are InWolwe, Process Miner, EMiT, Little Thumb, MiSon and ProM framework. Defining a common input format like Mining XML (MXML) was the first step towards the creation of a repository on which process mining researchers can test their algorithms. This XML format connects transactional systems such as workflow management systems, ERP systems, CRM systems, and case handling systems (Figure A.1) [3]. In principle, any system that registers events related to the execution of tasks for cases can use this tool independent format to store and exchange logs. The goal of using a single format is to reduce the implementation effort and to promote the use of these mining techniques in multiple contexts.

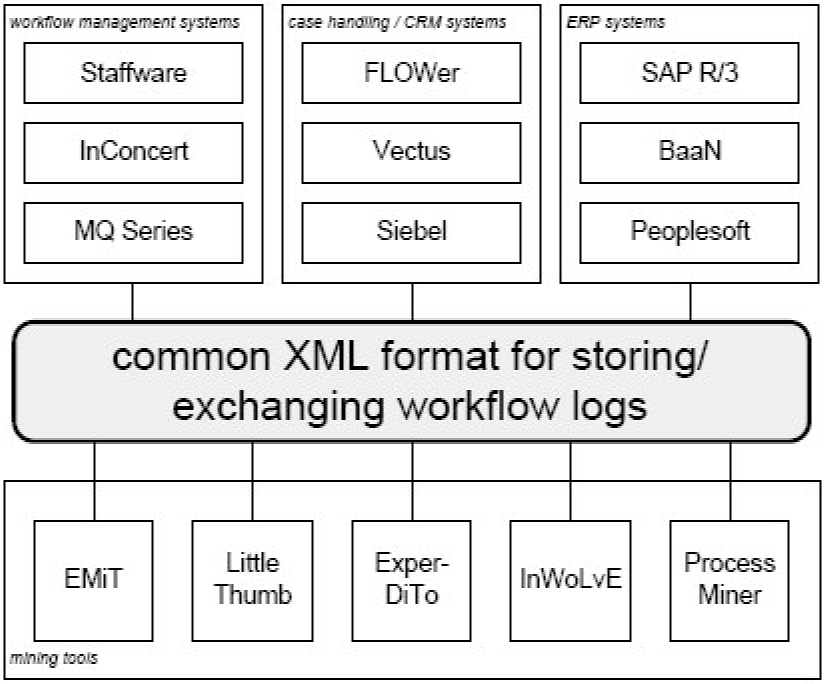


Figure A.1: The tool independent XML format connects transactional systems and the mining tools

The schema for this XML format is depicted in Figure A.2 [3]. The Document Type Definition on the basis of which Figure A.2 is derived is also shown in Figure A.3 [3]. As we can see in this figure, an event log (field WorkflowLog) has the execution of one or more processes (field Process), and optional information about the source program that generated the log (field Source) and additional data elements (field Data). Every process (field Process) has zero or more cases or process instances (field ProcessInstance). Similarly, every process instance has zero or more tasks (field AuditTrailEntry). Every task or audit trail entry (ATE) should at least have a name (field WorkflowModelElement) and an event type (field EventType). The event type determines the state of the tasks. There are 13 supported event types: schedule, assign, reassign, start, resume, suspend, autoskip, manualskip, withdraw, complete, ate abort, pi abort and unknown. The other task fields are optional. The Timestamp field supports the logging of time for the task. The Originator field records the person/system that performed the task. The Data field allows for more logging of additional information. Mapping the MXML format to the three mining perspectives, we see that the control-flow perspective mainly focuses on the WorkflowModelElement, the EventType and the Timestamp fields. The organizational perspective chiefly depends on the Originator field. The case perspective especially relies on the extra Data fields.

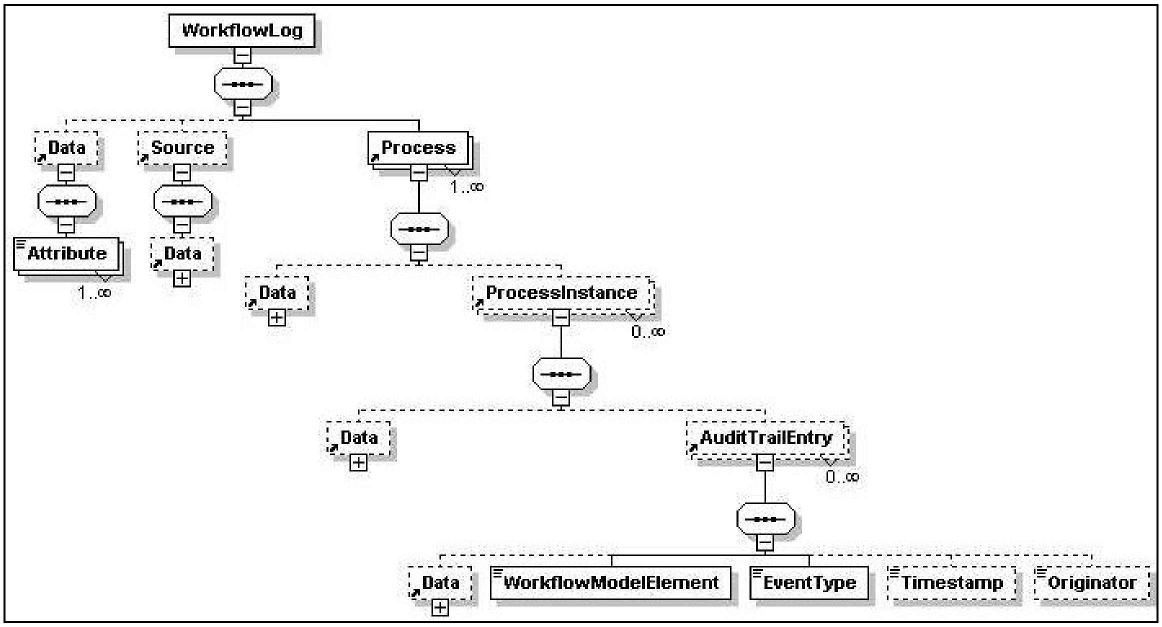
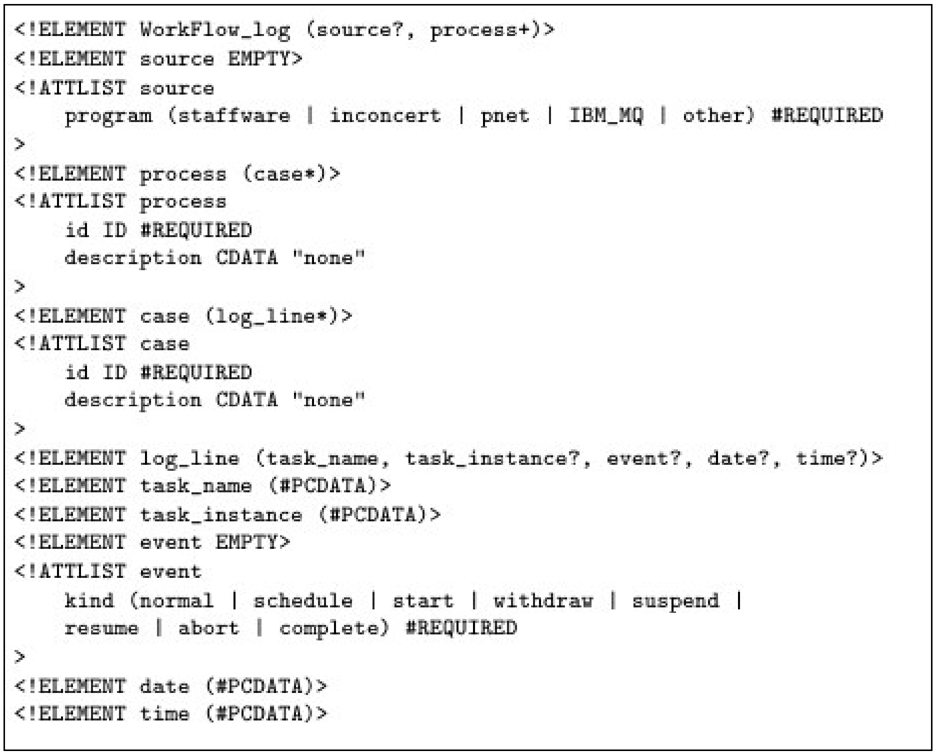


Figure A.2: Mining XML format schema



```
<!ELEMENT WorkFlow_log (source?, process+)>
<!ELEMENT source EMPTY>
<!ATTLIST source
    program (staffware | inconcert | pnet | IBM_MQ | other) #REQUIRED
>
<!ELEMENT process (case*)>
<!ATTLIST process
    id ID #REQUIRED
    description CDATA "none"
>
<!ELEMENT case (log_line*)>
<!ATTLIST case
    id ID #REQUIRED
    description CDATA "none"
>
<!ELEMENT log_line (task_name, task_instance?, event?, date?, time?)>
<!ELEMENT task_name (#PCDATA)>
<!ELEMENT task_instance (#PCDATA)>
<!ELEMENT event EMPTY>
<!ATTLIST event
    kind (normal | schedule | start | withdraw | suspend |
    resume | abort | complete) #REQUIRED
>
<!ELEMENT date (#PCDATA)>
<!ELEMENT time (#PCDATA)>
```

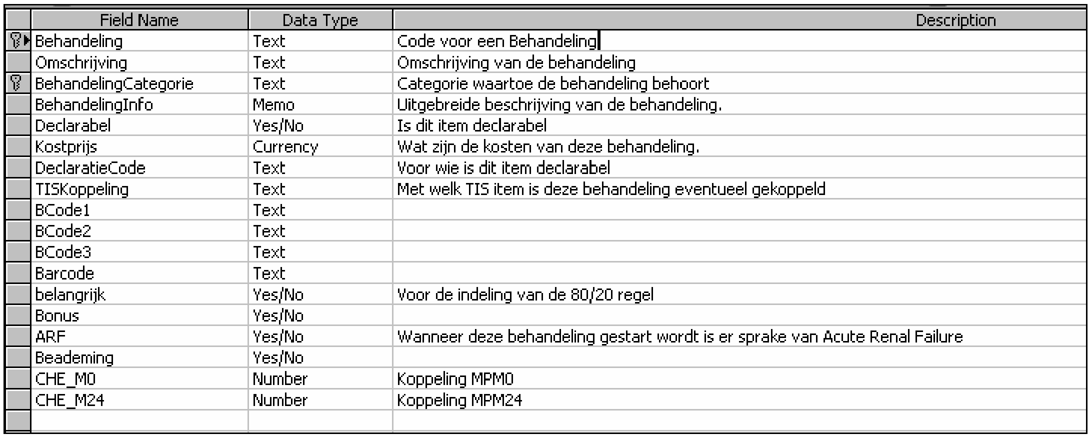
Figure A.3: The XML DTD for storing and exchanging workflow logs

B: Case study1

Data in Case study1 is organized in form of tables in MS-Access database. In this appendix, we provide all the important tables in this database. This gives us insights into what data is stored in the database and in what format. The data in these tables pertains to patient's general information, complications, treatments, measurements etc. For each table in the database we give the columns of the table (the structure of the tables in form of its fields), the data contained in these tables under these columns and the number of records in the table. Structure of any table includes: *field name, data type and description.* Field name is the name of the column in the data table, data type tells us what kind of data this field can hold i.e. number or text or Boolean etc. Description about the field is a short remark about the field. It is optional. At the end of the appendix we also mention some not-so important tables from the database without showing their structure and contents. It is also worth mentioning that information like patient number, specialist name, initials of patients and hospital staff etc. is anonymized for privacy reasons and the empty fields are not included in the figures.

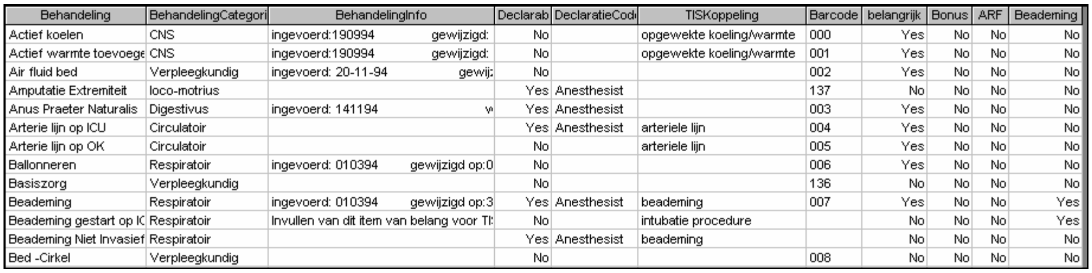
Behandeling

This table stores details about treatments that a patient may be given. It contains information about the treatment ID, treatment category, description, etc. These can be seen in the Figure B.1 which represents the design view of this table. The Figure B.2 shows a part of data contained in this table. The total number of records in this table is 173.



Field Name	Data Type	Description
Behandeling	Text	Code voor een Behandeling
Omschrijving	Text	Omschrijving van de behandeling
BehandelingCategorie	Text	Categorie waartoe de behandeling behoort
BehandelingInfo	Memo	Uitgebreide beschrijving van de behandeling.
Declarabel	Yes/No	Is dit item declarabel
Kostprijs	Currency	Wat zijn de kosten van deze behandeling.
DeclaratieCode	Text	Voor wie is dit item declarabel
TISKoppeling	Text	Met welk TIS item is deze behandeling eventueel gekoppeld
BCode1	Text	
BCode2	Text	
BCode3	Text	
Barcode	Text	
belangrijk	Yes/No	Voor de indeling van de 80/20 regel
Bonus	Yes/No	
ARF	Yes/No	Wanneer deze behandeling gestart wordt is er sprake van Acute Renal Failure
Beademing	Yes/No	
CHE_M0	Number	Koppeling MPM0
CHE_M24	Number	Koppeling MPM24

Figure B.1: Structure of the *Behandeling* (Treatment) table

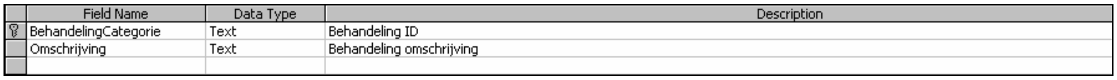


Behandeling	BehandelingCategori	BehandelingInfo	Declarab	DeclaratieCod	TISKoppeling	Barcode	belangrijk	Bonus	ARF	Beademing
Actief koelen	CNS	ingevoerd:190994 gewijzigd:	No		opgewekte koeling/warmte	000	Yes	No	No	No
Actief warmte toevoege	CNS	ingevoerd:190994 gewijzigd:	No		opgewekte koeling/warmte	001	Yes	No	No	No
Air fluid bed	Verpleegkundig	ingevoerd: 20-11-94 gewij:	No			002	Yes	No	No	No
Amputatie Extremiteit	loco-motrius		Yes	Anesthesist		137	No	No	No	No
Anus Praeter Naturalis	Digestivus	ingevoerd: 141194 w	Yes	Anesthesist		003	Yes	No	No	No
Arterie lijn op ICU	Circulatoir		Yes	Anesthesist	arteriele lijn	004	Yes	No	No	No
Arterie lijn op OK	Circulatoir		No		arteriele lijn	005	Yes	No	No	No
Ballonneren	Respiratoir	ingevoerd: 010394 gewijzigd op:0	No			006	Yes	No	No	No
Basiszorg	Verpleegkundig		No			136	No	No	No	No
Beademing	Respiratoir	ingevoerd: 010394 gewijzigd op:3	Yes	Anesthesist	beademing	007	Yes	No	No	Yes
Beademing gestart op IC	Respiratoir	Invullen van dit item van belang voor TI:	No		intubatie procedure		No	No	No	Yes
Beademing Niet Invasief	Respiratoir		Yes	Anesthesist	beademing		No	No	No	No
Bed -Cirkel	Verpleegkundig		No			008	No	No	No	No

Figure B.2: Data in the *Behandeling* table

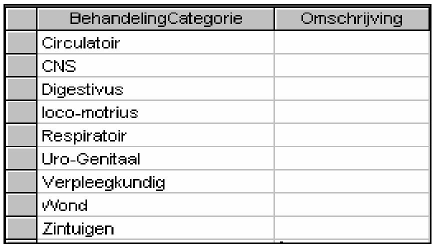
BehandelingCategorie

This table stores details about treatment categories for the treatments that a patient may be given. Figure B.4 shows these categories. The total number of records in this table is 9. We can see that there is no information in the *omschrijiving* field.



	Field Name	Data Type	Description
🔑	BehandelingCategorie	Text	Behandeling ID
	Omschrijving	Text	Behandeling omschrijving

Figure B.3: Structure of the *BehandelingCategorie* (Treatment category) table

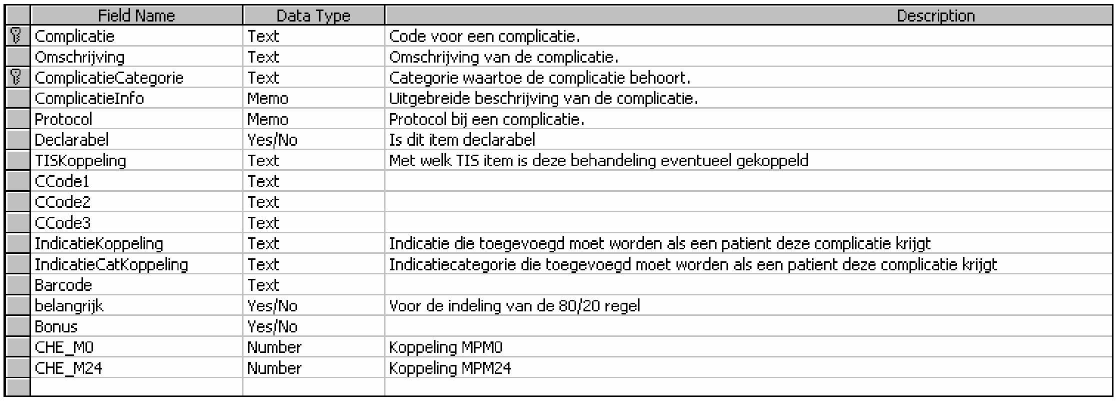


BehandelingCategorie	Omschrijving
Circulatoir	
CNS	
Digestivus	
loco-motrius	
Respiratoir	
Uro-Genitaal	
Verpleegkundig	
Wond	
Zintuigen	

Figure B.4: Data in the *BehandelingCategorie* table

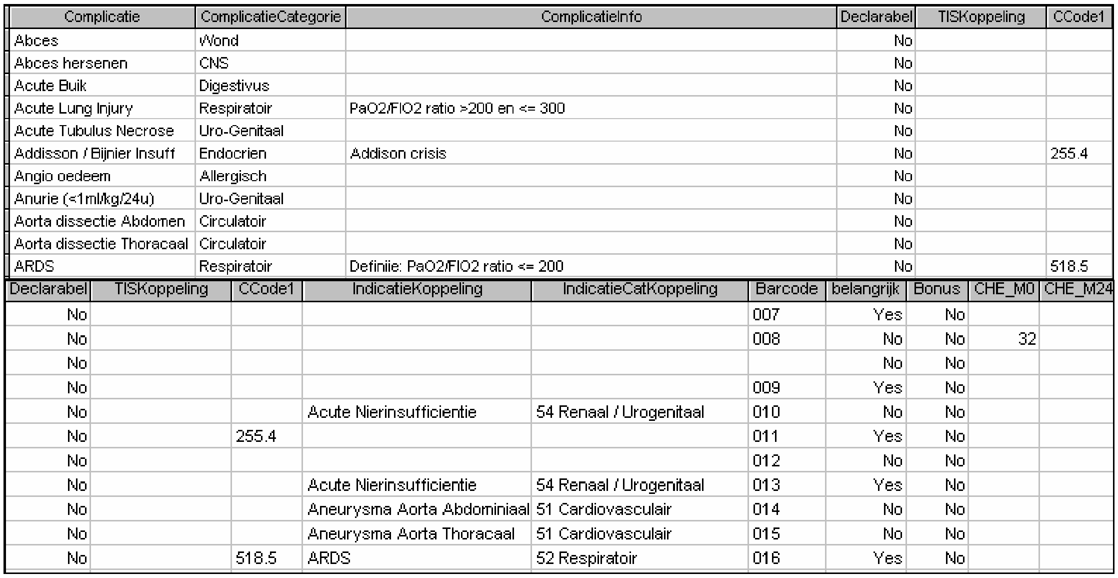
Complicatie

This table stores details about complications that a patient may suffer from. It contains information about the complication ID, complication category, description, etc. Figure B.5 shows these data attributes, and the next figures show a part of data contained in these fields. The total number of records in this table is 155. As seen, we have combined data from various columns in the same figure (Figure B.6). The grey shaded row shows the column/field names.



	Field Name	Data Type	Description
🔑	Complicatie	Text	Code voor een complicatie.
	Omschrijving	Text	Omschrijving van de complicatie.
🔑	ComplicatieCategorie	Text	Categorie waartoe de complicatie behoort.
	ComplicatieInfo	Memo	Uitgebreide beschrijving van de complicatie.
	Protocol	Memo	Protocol bij een complicatie.
	Declarabel	Yes/No	Is dit item declarabel
	TISKoppeling	Text	Met welk TIS item is deze behandeling eventueel gekoppeld
	CCode1	Text	
	CCode2	Text	
	CCode3	Text	
	IndicatieKoppeling	Text	Indicatie die toegevoegd moet worden als een patient deze complicatie krijgt
	IndicatieCatKoppeling	Text	Indicatiecategorie die toegevoegd moet worden als een patient deze complicatie krijgt
	Barcode	Text	
	belangrijk	Yes/No	Voor de indeling van de 80/20 regel
	Bonus	Yes/No	
	CHE_M0	Number	Koppeling MPM0
	CHE_M24	Number	Koppeling MPM24

Figure B.5: Structure of the *Complicatie* (Complication) table



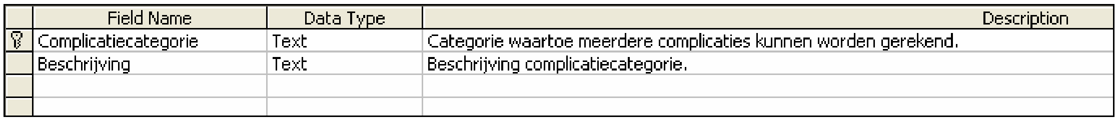
Complicatie	ComplicatieCategorie	ComplicatieInfo	Declarabel	TISKoppeling	CCode1
Abces	Wond		No		
Abces hersenen	CNS		No		
Acute Buik	Digestivus		No		
Acute Lung Injury	Respiratoir	PaO2/FIO2 ratio >200 en <= 300	No		
Acute Tubulus Necrose	Uro-Genitaal		No		
Addisson / Bijnier Insuff	Endocrien	Addison crisis	No		255.4
Angio oedeem	Allergisch		No		
Anurie (<1ml/kg/24u)	Uro-Genitaal		No		
Aorta dissectie Abdomen	Circulatoir		No		
Aorta dissectie Thoracaal	Circulatoir		No		
ARDS	Respiratoir	Definiie: PaO2/FIO2 ratio <= 200	No		518.5

Declarabel	TISKoppeling	CCode1	IndicatieKoppeling	IndicatieCatKoppeling	Barcode	belangrijk	Bonus	CHE_M0	CHE_M24
No					007	Yes	No		
No					008	No	No	32	
No						No	No		
No					009	Yes	No		
No			Acute Nierinsufficientie	54 Renaal / Urogenitaal	010	No	No		
No		255.4			011	Yes	No		
No					012	No	No		
No			Acute Nierinsufficientie	54 Renaal / Urogenitaal	013	Yes	No		
No			Aneurysma Aorta Abdominiaal	51 Cardiovasculair	014	No	No		
No			Aneurysma Aorta Thoracaal	51 Cardiovasculair	015	No	No		
No		518.5	ARDS	52 Respiratoir	016	Yes	No		

Figure B.6: Data in the *Complicatie* table

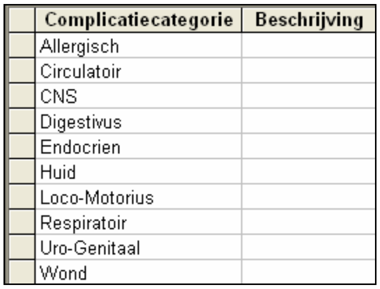
ComplicatieCategorie

This table provides the categories of complications. Each category pertains to specific type of complications. For example, complications related to respiration and similar problems fall under the category 'respiratoir'. The total number of records in this table is 10.



	Field Name	Data Type	Description
🔑	Complicatiecategorie	Text	Categorie waartoe meerdere complicaties kunnen worden gerekend.
	Beschrijving	Text	Beschrijving complicatiecategorie.

Figure B.7: Structure of the *ComplicatieCategorie* table

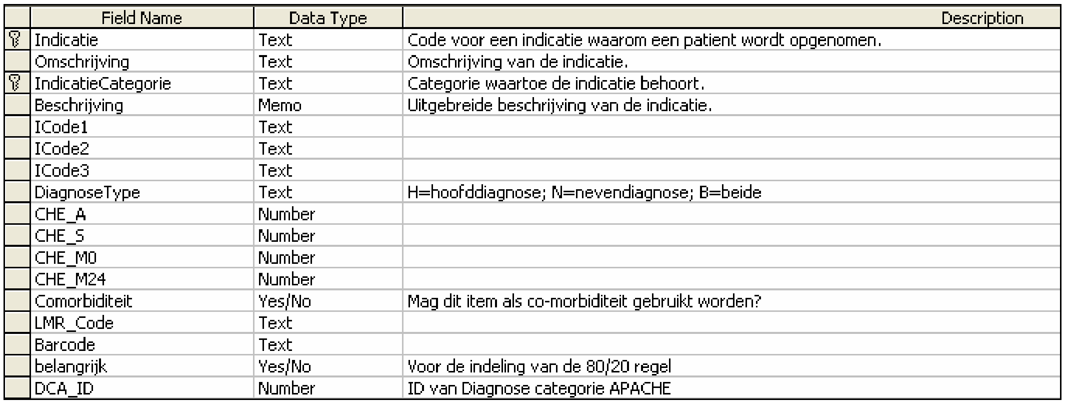


Complicatiecategorie	Beschrijving
Allergisch	
Circulatoir	
CNS	
Digestivus	
Endocrien	
Huid	
Loco-Motorius	
Respiratoir	
Uro-Genitaal	
Wond	

Figure B.8: Data in the *ComplicatieCategorie* table

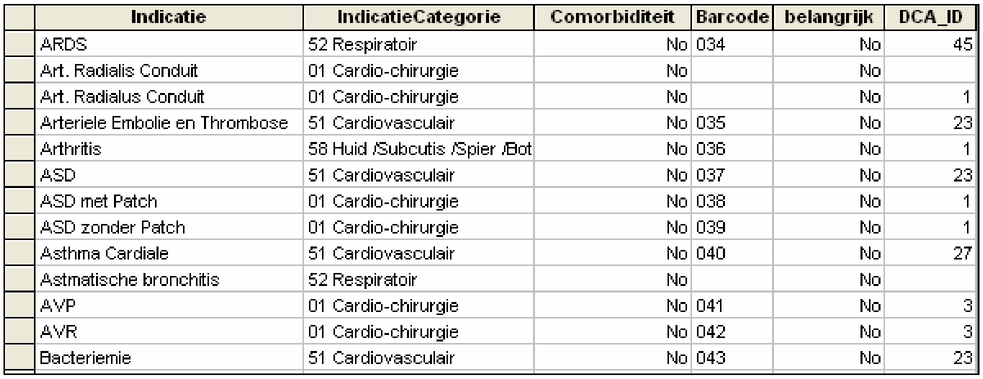
Indicatie

This table is the master table containing records for possible indications a patient shows when he suffers from a complication. The table includes data under the headings like *Indication category, diagnosis type etc.* The total number of records in this table is 485.



	Field Name	Data Type	Description
🔑	Indicatie	Text	Code voor een indicatie waarom een patient wordt opgenomen.
	Omschrijving	Text	Omschrijving van de indicatie.
🔑	IndicatieCategorie	Text	Categorie waartoe de indicatie behoort.
	Beschrijving	Memo	Uitgebreide beschrijving van de indicatie.
	ICode1	Text	
	ICode2	Text	
	ICode3	Text	
	DiagnoseType	Text	H=hoofddiagnose; N=nevendiagnose; B=beide
	CHE_A	Number	
	CHE_S	Number	
	CHE_M0	Number	
	CHE_M24	Number	
	Comorbiditeit	Yes/No	Mag dit item als co-morbiditeit gebruikt worden?
	LMR_Code	Text	
	Barcode	Text	
	belangrijk	Yes/No	Voor de indeling van de 80/20 regel
	DCA_ID	Number	ID van Diagnose categorie APACHE

Figure B.9: Structure of the *Indicatie* table

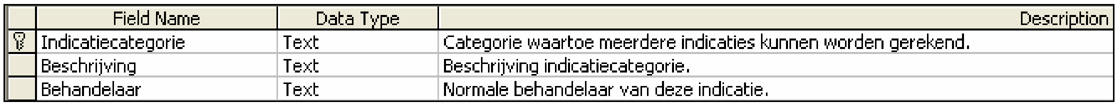


Indicatie	IndicatieCategorie	Comorbiditeit	Barcode	belangrijk	DCA_ID
ARDS	52 Respiratoir	No	034	No	45
Art. Radialis Conduit	01 Cardio-chirurgie	No		No	
Art. Radialus Conduit	01 Cardio-chirurgie	No		No	1
Arteriele Embolie en Thrombose	51 Cardiovasculair	No	035	No	23
Arthritis	58 Huid /Subcutis /Spier /Bot	No	036	No	1
ASD	51 Cardiovasculair	No	037	No	23
ASD met Patch	01 Cardio-chirurgie	No	038	No	1
ASD zonder Patch	01 Cardio-chirurgie	No	039	No	1
Asthma Cardiale	51 Cardiovasculair	No	040	No	27
Astmatische bronchitis	52 Respiratoir	No		No	
AVP	01 Cardio-chirurgie	No	041	No	3
AVR	01 Cardio-chirurgie	No	042	No	3
Bacteriemie	51 Cardiovasculair	No	043	No	23

Figure B.10: Data in the *Indicatie* table

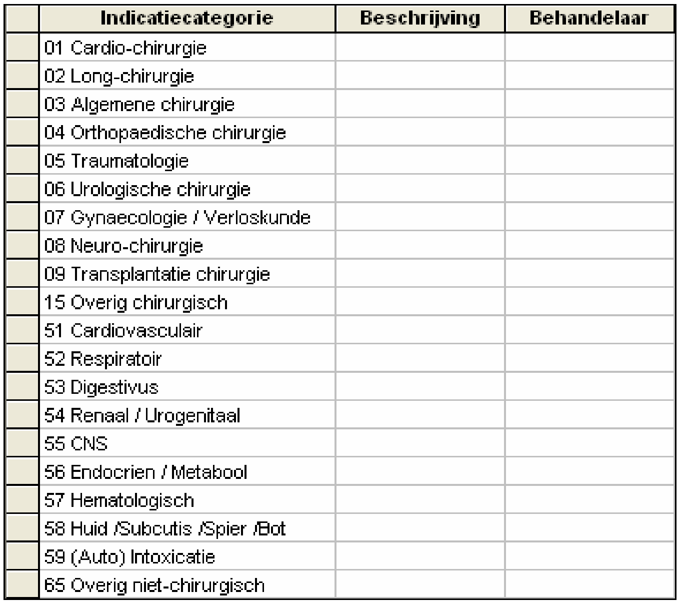
IndicatieCategorie

Just like the table *ComplicatieCategorie,* this table provides the categories of indiactions. Each category pertains to specific type of indications/major symptoms. For example, complications related to heart problems fall under the category 'Cardiovasculair. The total number of records in this table is 20.



	Field Name	Data Type	Description
🔑	Indicatiecategorie	Text	Categorie waartoe meerdere indicaties kunnen worden gerekend.
	Beschrijving	Text	Beschrijving indicatiecategorie.
	Behandelaar	Text	Normale behandelaar van deze indicatie.

Figure B.11: Structure of the *IndicatieCategorie* table

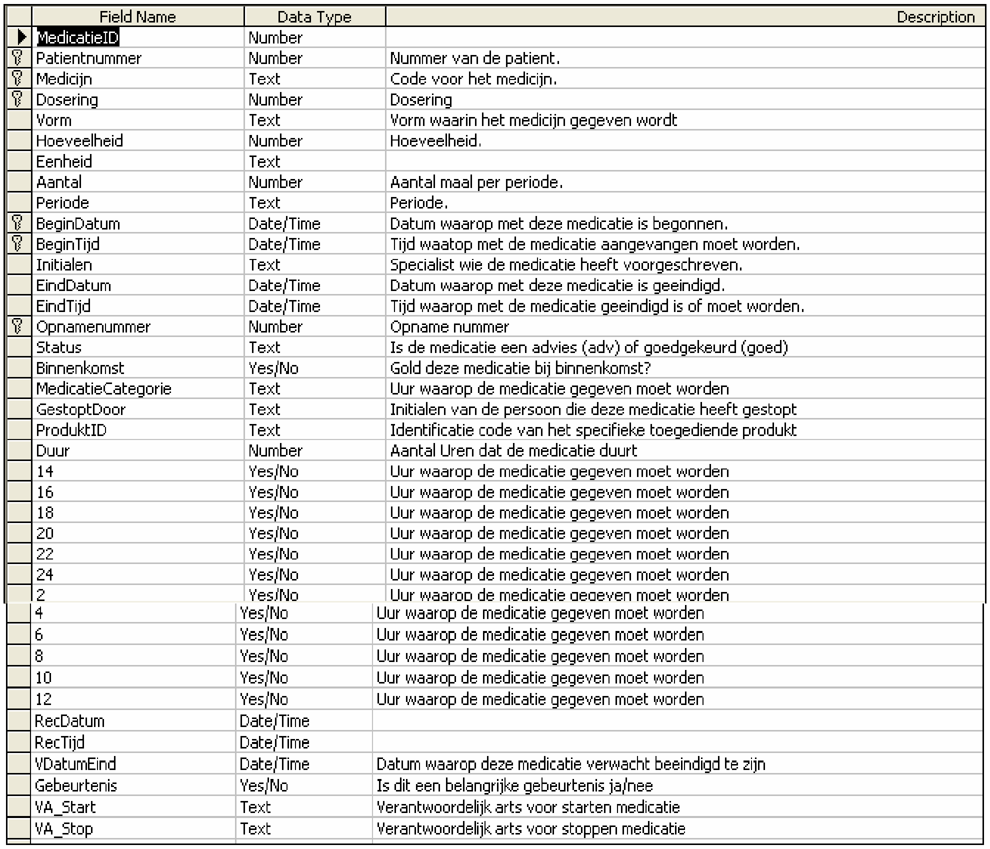


Indicatiecategorie	Beschrijving	Behandelaar
01 Cardio-chirurgie		
02 Long-chirurgie		
03 Algemene chirurgie		
04 Orthopaedische chirurgie		
05 Traumatologie		
06 Urologische chirurgie		
07 Gynaecologie / Verloskunde		
08 Neuro-chirurgie		
09 Transplantatie chirurgie		
15 Overig chirurgisch		
51 Cardiovasculair		
52 Respiratoir		
53 Digestivus		
54 Renaal / Urogenitaal		
55 CNS		
56 Endocrien / Metabool		
57 Hematologisch		
58 Huid /Subcutis /Spier /Bot		
59 (Auto) Intoxicatie		
65 Overig niet-chirurgisch		

Figure B.12: Data in the *IndicatieCategorie* table

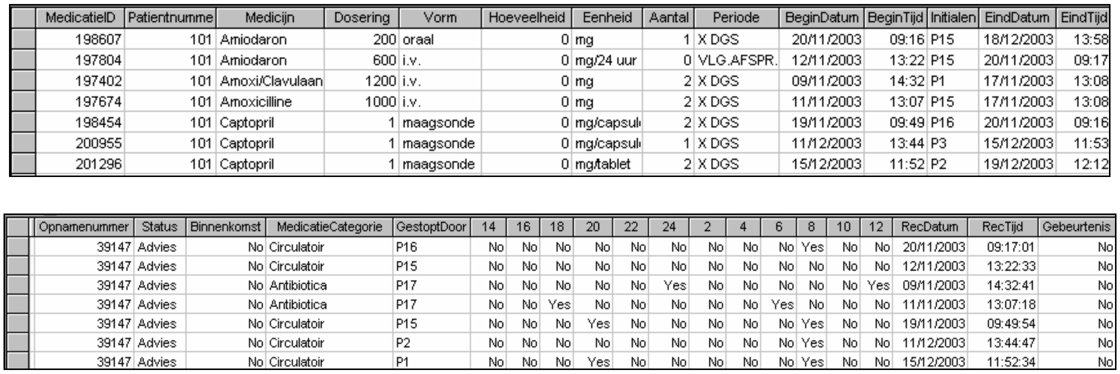
Medicatie

This table contains records for medication given to patients. It includes information like a description why the medicine is given to the patient, the time period for which the medicine is given, the patient number indicating the patient to which this record belongs to etc. The details of the medicine can be found in the table *Medicijn* explained next. The total number of records in this table is 39586. Figure B.14 shows data contained in various columns. Each grey shaded row should be read as column names. The table should be read as: each row under the two grey shaded rows belong to *MedicatieID*, it means the rows under the second grey shaded row pertains respectively to rows below the first grey shaded row.



Field Name	Data Type	Description
MedicatieID	Number	
Patientnummer	Number	Nummer van de patient.
Medicijn	Text	Code voor het medicijn.
Dosering	Number	Dosering
Vorm	Text	Vorm waarin het medicijn gegeven wordt
Hoeveelheid	Number	Hoeveelheid.
Eenheid	Text	
Aantal	Number	Aantal maal per periode.
Periode	Text	Periode.
BeginDatum	Date/Time	Datum waarop met deze medicatie is begonnen.
BeginTijd	Date/Time	Tijd waatop met de medicatie aangevangen moet worden.
Initialen	Text	Specialist wie de medicatie heeft voorgeschreven.
EindDatum	Date/Time	Datum waarop met deze medicatie is geeindigd.
EindTijd	Date/Time	Tijd waarop met de medicatie geeindigd is of moet worden.
Opnamenummer	Number	Opname nummer
Status	Text	Is de medicatie een advies (adv) of goedgekeurd (goed)
Binnenkomst	Yes/No	Gold deze medicatie bij binnenkomst?
MedicatieCategorie	Text	Uur waarop de medicatie gegeven moet worden
GestoptDoor	Text	Initialen van de persoon die deze medicatie heeft gestopt
ProduktID	Text	Identificatie code van het specifieke toegediende produkt
Duur	Number	Aantal Uren dat de medicatie duurt
14	Yes/No	Uur waarop de medicatie gegeven moet worden
16	Yes/No	Uur waarop de medicatie gegeven moet worden
18	Yes/No	Uur waarop de medicatie gegeven moet worden
20	Yes/No	Uur waarop de medicatie gegeven moet worden
22	Yes/No	Uur waarop de medicatie gegeven moet worden
24	Yes/No	Uur waarop de medicatie gegeven moet worden
2	Yes/No	Uur waarop de medicatie gegeven moet worden
4	Yes/No	Uur waarop de medicatie gegeven moet worden
6	Yes/No	Uur waarop de medicatie gegeven moet worden
8	Yes/No	Uur waarop de medicatie gegeven moet worden
10	Yes/No	Uur waarop de medicatie gegeven moet worden
12	Yes/No	Uur waarop de medicatie gegeven moet worden
RecDatum	Date/Time	
RecTijd	Date/Time	
VDatumEind	Date/Time	Datum waarop deze medicatie verwacht beeindigd te zijn
Gebeurtenis	Yes/No	Is dit een belangrijke gebeurtenis ja/nee
VA_Start	Text	Verantwoordelijk arts voor starten medicatie
VA_Stop	Text	Verantwoordelijk arts voor stoppen medicatie

Figure B.13: Structure of the *Medicatie* table



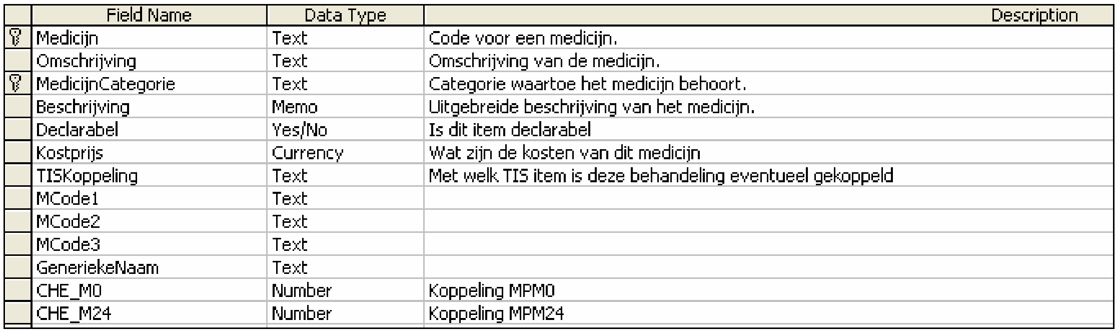
MedicatieID	Patientnumme	Medicijn	Dosering	Vorm	Hoeveelheid	Eenheid	Aantal	Periode	BeginDatum	BeginTijd	Initialen	EindDatum	EindTijd
198607	101	Amiodaron	200	oraal	0	mg	1	X DGS	20/11/2003	09:16	P15	18/12/2003	13:58
197804	101	Amiodaron	600	i.v.	0	mg/24 uur	0	VLG.AFSPR.	12/11/2003	13:22	P15	20/11/2003	09:17
197402	101	Amoxi/Clavulaan	1200	i.v.	0	mg	2	X DGS	09/11/2003	14:32	P1	17/11/2003	13:08
197674	101	Amoxicilline	1000	i.v.	0	mg	2	X DGS	11/11/2003	13:07	P15	17/11/2003	13:08
198454	101	Captopril	1	maagsonde	0	mg/capsul	2	X DGS	19/11/2003	09:49	P16	20/11/2003	09:16
200955	101	Captopril	1	maagsonde	0	mg/capsul	1	X DGS	11/12/2003	13:44	P3	15/12/2003	11:53
201296	101	Captopril	1	maagsonde	0	mg/tablet	2	X DGS	15/12/2003	11:52	P2	19/12/2003	12:12

Opnamenummer	Status	Binnenkomst	MedicatieCategorie	GestoptDoor	14	16	18	20	22	24	2	4	6	8	10	12	RecDatum	RecTijd	Gebeurtenis
39147	Advies	No	Circulatoir	P16	No	No	No	No	No	No	No	No	No	Yes	No	No	20/11/2003	09:17:01	No
39147	Advies	No	Circulatoir	P15	No	No	No	No	No	No	No	No	No	No	No	No	12/11/2003	13:22:33	No
39147	Advies	No	Antibiotica	P17	No	No	No	No	No	Yes	No	No	No	No	No	Yes	09/11/2003	14:32:41	No
39147	Advies	No	Antibiotica	P17	No	No	Yes	No	No	No	No	No	Yes	No	No	No	11/11/2003	13:07:18	No
39147	Advies	No	Circulatoir	P15	No	No	No	Yes	No	No	No	No	No	Yes	No	No	19/11/2003	09:49:54	No
39147	Advies	No	Circulatoir	P2	No	No	No	No	No	No	No	No	No	Yes	No	No	11/12/2003	13:44:47	No
39147	Advies	No	Circulatoir	P1	No	No	No	Yes	No	No	No	No	No	Yes	No	No	15/12/2003	11:52:34	No

Figure B.14: Data in the *Medicatie* table

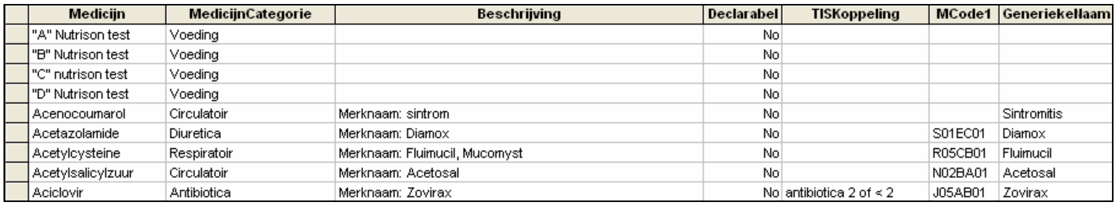
Medicijn

This table stores details about medicines, which are given to the patients. The table *Medicatie* stores details about medicines and the patients receiving them, this table gives the information about the medicines alone. It includes information like the medicine category, cost price of the drug, description etc. The total number of records in this table is 505.



Field Name	Data Type	Description
Medicijn	Text	Code voor een medicijn.
Omschrijving	Text	Omschrijving van de medicijn.
MedicijnCategorie	Text	Categorie waartoe het medicijn behoort.
Beschrijving	Memo	Uitgebreide beschrijving van het medicijn.
Declarabel	Yes/No	Is dit item declarabel
Kostprijs	Currency	Wat zijn de kosten van dit medicijn
TISKoppeling	Text	Met welk TIS item is deze behandeling eventueel gekoppeld
MCode1	Text	
MCode2	Text	
MCode3	Text	
GeneriekeNaam	Text	
CHE_M0	Number	Koppeling MPM0
CHE_M24	Number	Koppeling MPM24

Figure B.15: Structure of the *Medicijn* table

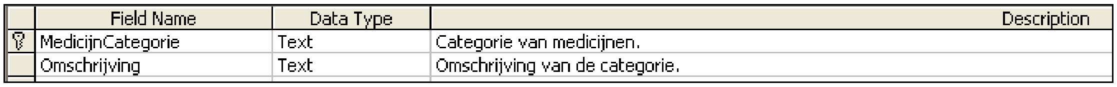


Medicijn	MedicijnCategorie	Beschrijving	Declarabel	TISKoppeling	MCode1	GeneriekeNaam
"A" Nutrison test	Voeding		No			
"B" Nutrison test	Voeding		No			
"C" nutrison test	Voeding		No			
"D" Nutrison test	Voeding		No			
Acenocoumarol	Circulatoir	Merknaam: sintrom	No			Sintromitis
Acetazolamide	Diuretica	Merknaam: Diamox	No		S01EC01	Diamox
Acetylcysteine	Respiratoir	Merknaam: Fluimucil, Mucomyst	No		R05CB01	Fluimucil
Acetylsalicylzuur	Circulatoir	Merknaam: Acetosal	No		N02BA01	Acetosal
Aciclovir	Antibiotica	Merknaam: Zovirax	No	antibiotica 2 of < 2	J05AB01	Zovirax

Figure B.16: Data in the *Medicijn* table

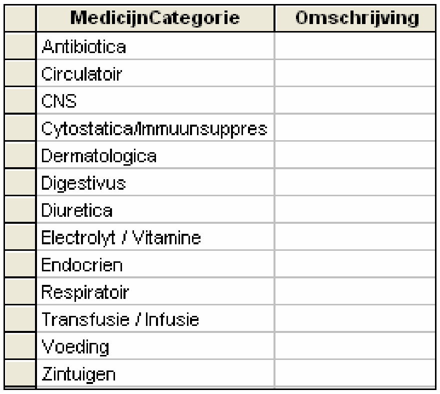
MedicijnCategorie

The medicine category information mentioned in the table *Medicijn* comes from this master table. The total number of records in this table is 505.



Field Name	Data Type	Description
MedicijnCategorie	Text	Categorie van medicijnen.
Omschrijving	Text	Omschrijving van de categorie.

Figure B.17: Structure of the *MedicijnCategorie* table

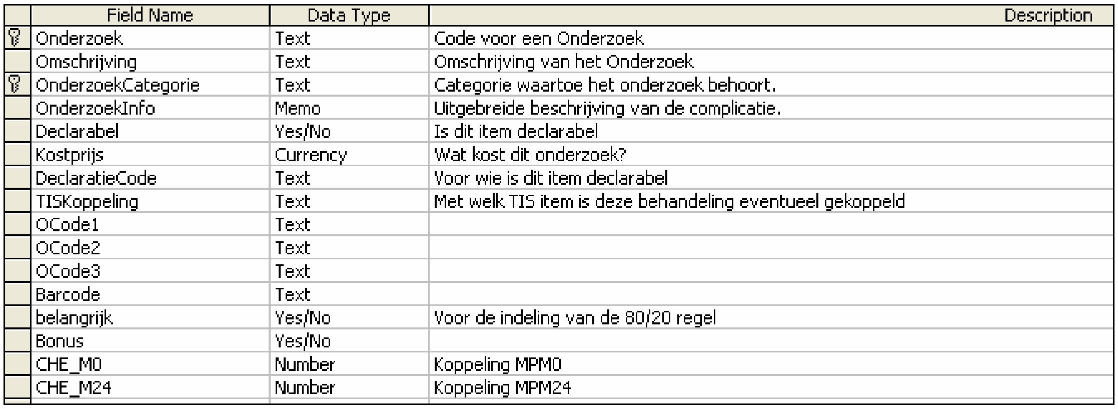


MedicijnCategorie	Omschrijving
Antibiotica	
Circulatoir	
CNS	
Cytostatica/Immuunsuppres	
Dermatologica	
Digestivus	
Diuretica	
Electrolyt / Vitamine	
Endocrien	
Respiratoir	
Transfusie / Infusie	
Voeding	
Zintuigen	

Figure B.18: Data in the MedicijnCategorie table

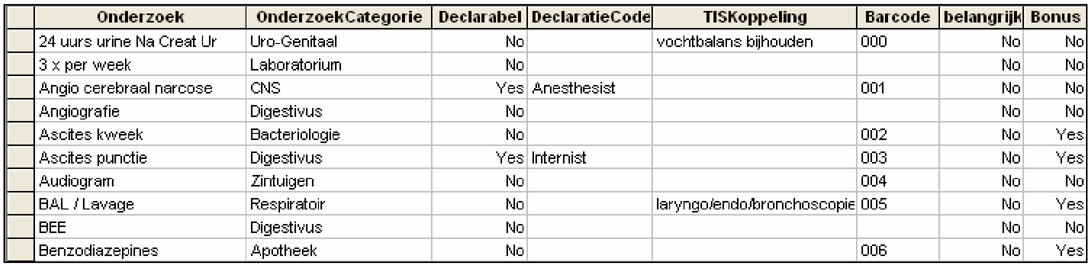
Onderzoek

This table stores details about the measurements or medical tests which can be done on the patients. This table is a generic table giving this information without relating it to the records about which patient undergoes what measurement. It includes information like the (onderzoek) measurement category, cost price of the test, description etc. The total number of records in this table is 144.



	Field Name	Data Type	Description
🔑	Onderzoek	Text	Code voor een Onderzoek
	Omschrijving	Text	Omschrijving van het Onderzoek
🔑	OnderzoekCategorie	Text	Categorie waartoe het onderzoek behoort.
	OnderzoekInfo	Memo	Uitgebreide beschrijving van de complicatie.
	Declarabel	Yes/No	Is dit item declarabel
	Kostprijs	Currency	Wat kost dit onderzoek?
	DeclaratieCode	Text	Voor wie is dit item declarabel
	TISKoppeling	Text	Met welk TIS item is deze behandeling eventueel gekoppeld
	OCode1	Text	
	OCode2	Text	
	OCode3	Text	
	Barcode	Text	
	belangrijk	Yes/No	Voor de indeling van de 80/20 regel
	Bonus	Yes/No	
	CHE_M0	Number	Koppeling MPM0
	CHE_M24	Number	Koppeling MPM24

Figure B.19: Structure of the Onderzoek table

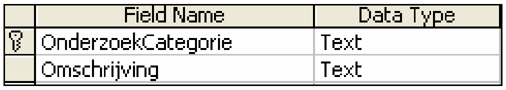


Onderzoek	OnderzoekCategorie	Declarabel	DeclaratieCode	TISKoppeling	Barcode	belangrijk	Bonus
24 uurs urine Na Creat Ur	Uro-Genitaal	No		vochtbalans bijhouden	000	No	No
3 x per week	Laboratorium	No				No	No
Angio cerebraal narcose	CNS	Yes	Anesthesist		001	No	No
Angiografie	Digestivus	No				No	No
Ascites kweek	Bacteriologie	No			002	No	Yes
Ascites punctie	Digestivus	Yes	Internist		003	No	Yes
Audiogram	Zintuigen	No			004	No	No
BAL / Lavage	Respiratoir	No		laryngo/endo/bronchoscopie	005	No	Yes
BEE	Digestivus	No				No	No
Benzodiazepines	Apotheek	No			006	No	Yes

Figure B.20: Data in the Onderzoek table

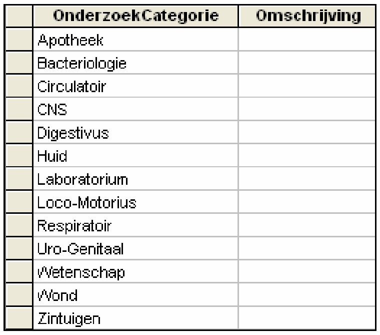
OnderzoekCategorie

The measurement category information mentioned in the table *Onderzoek* comes from this master table. The total number of records in this table is 13.



	Field Name	Data Type
🔑	OnderzoekCategorie	Text
	Omschrijving	Text

Figure B.21: Structure of the OnderzoekCategorie table

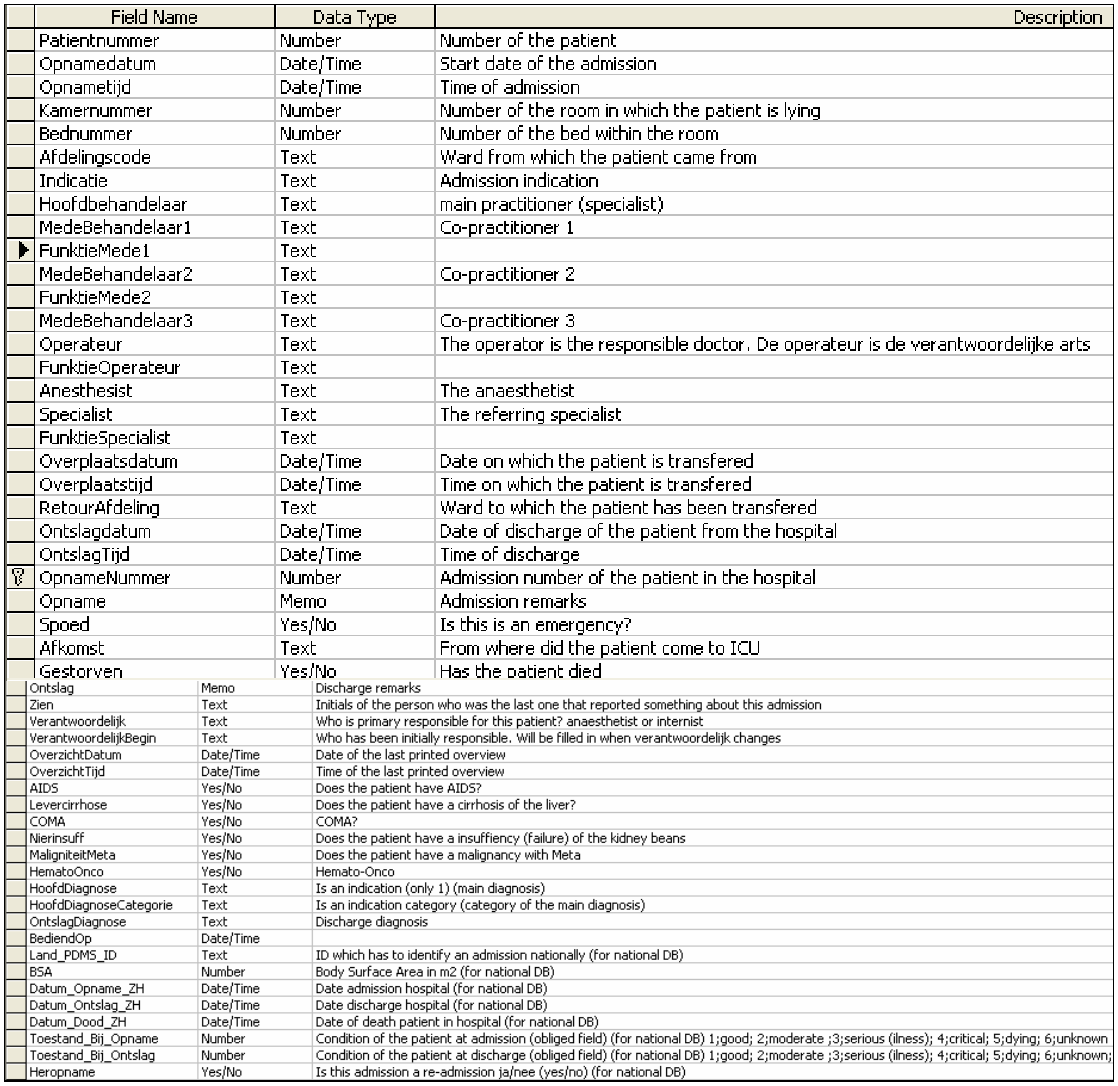


OnderzoekCategorie	Omschrijving
Apotheek	
Bacteriologie	
Circulatoir	
CNS	
Digestivus	
Huid	
Laboratorium	
Loco-Motorius	
Respiratoir	
Uro-Genitaal	
Wetenschap	
Wond	
Zintuigen	

Figure B.22: Data in the OnderzoekCategorie table

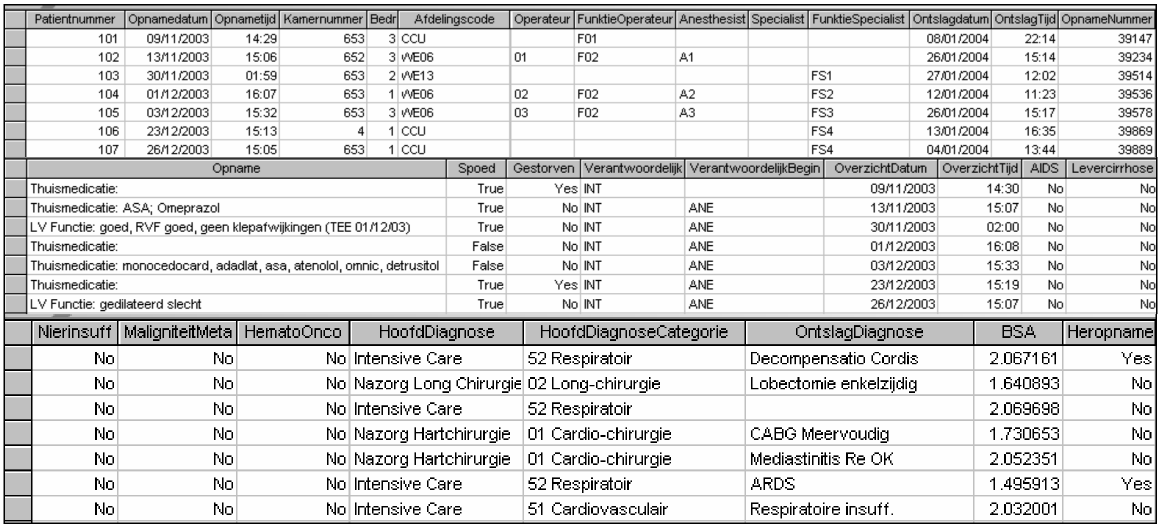
Opname

Opname is the table storing complete information about a patient’s admission to the hospital. It includes his admission details like the date and time of his admission, a preliminary reason for his admission, medical staff (doctors, practitioners, anaesthetist etc.) responsible for him, discharge details etc. The total number of records in this table is 2964. . Figure B.24 shows data contained in various columns. Each grey shaded row should be read as column names. The table should be read as: each row under the two grey shaded rows belong to the record of the patient with *Patientnummer*, it means the rows under the second and third grey shaded row pertains respectively to rows below the first grey shaded row.



Field Name	Data Type	Description
Patientnummer	Number	Number of the patient
Opnamedatum	Date/Time	Start date of the admission
Opnametijd	Date/Time	Time of admission
Kamernummer	Number	Number of the room in which the patient is lying
Bednummer	Number	Number of the bed within the room
Afdelingscode	Text	Ward from which the patient came from
Indicatie	Text	Admission indication
Hoofdbehandelaar	Text	main practitioner (specialist)
MedeBehandelaar1	Text	Co-practitioner 1
FunktieMede1	Text	
MedeBehandelaar2	Text	Co-practitioner 2
FunktieMede2	Text	
MedeBehandelaar3	Text	Co-practitioner 3
Operateur	Text	The operator is the responsible doctor. De operateur is de verantwoordelijke arts
FunktieOperateur	Text	
Anesthesist	Text	The anaesthetist
Specialist	Text	The referring specialist
FunktieSpecialist	Text	
Overplaatsdatum	Date/Time	Date on which the patient is transfered
Overplaatstijd	Date/Time	Time on which the patient is transfered
RetourAfdeling	Text	Ward to which the patient has been transfered
Ontslagdatum	Date/Time	Date of discharge of the patient from the hospital
OntslagTijd	Date/Time	Time of discharge
OpnameNummer	Number	Admission number of the patient in the hospital
Opname	Memo	Admission remarks
Spoed	Yes/No	Is this is an emergency?
Afkomst	Text	From where did the patient come to ICU
Gestorven	Yes/No	Has the patient died
Ontslag	Memo	Discharge remarks
Zien	Text	Initials of the person who was the last one that reported something about this admission
Verantwoordelijk	Text	Who is primary responsible for this patient? anaesthetist or internist
VerantwoordelijkBegin	Text	Who has been initially responsible. Will be filled in when verantwoordelijk changes
OverzichtDatum	Date/Time	Date of the last printed overview
OverzichtTijd	Date/Time	Time of the last printed overview
AIDS	Yes/No	Does the patient have AIDS?
Levercirrhose	Yes/No	Does the patient have a cirrhosis of the liver?
COMA	Yes/No	COMA?
Nierinsuff	Yes/No	Does the patient have a insuffiency (failure) of the kidney beans
MaligniteitMeta	Yes/No	Does the patient have a malignancy with Meta
HematoOnco	Yes/No	Hemato-Onco
HoofdDiagnose	Text	Is an indication (only 1) (main diagnosis)
HoofdDiagnoseCategorie	Text	Is an indication category (category of the main diagnosis)
OntslagDiagnose	Text	Discharge diagnosis
BediendOp	Date/Time	
Land_PDMS_ID	Text	ID which has to identify an admission nationally (for national DB)
BSA	Number	Body Surface Area in m2 (for national DB)
Datum_Opname_ZH	Date/Time	Date admission hospital (for national DB)
Datum_Ontslag_ZH	Date/Time	Date discharge hospital (for national DB)
Datum_Dood_ZH	Date/Time	Date of death patient in hospital (for national DB)
Toestand_Bij_Opname	Number	Condition of the patient at admission (obliged field) (for national DB) 1;good; 2;moderate ;3;serious (ilness); 4;critical; 5;dying; 6;unknown
Toestand_Bij_Ontslag	Number	Condition of the patient at discharge (obliged field) (for national DB) 1;good; 2;moderate ;3;serious (ilness); 4;critical; 5;dying; 6;unknown;
Heropname	Yes/No	Is this admission a re-admission ja/nee (yes/no) (for national DB)

Figure B.23: Structure of the Opname table



Patientnummer	Opnamedatum	Opnametijd	Kamernummer	Bedr	Afdelingscode	Operateur	FunktieOperateur	Anesthesist	Specialist	FunktieSpecialist	Ontslagdatum	OntslagTijd	OpnameNummer
101	09/11/2003	14:29	653	3	CCU		F01				08/01/2004	22:14	39147
102	13/11/2003	15:06	652	3	WE06	01	F02	A1			26/01/2004	15:14	39234
103	30/11/2003	01:59	653	2	WE13					FS1	27/01/2004	12:02	39514
104	01/12/2003	16:07	653	1	WE06	02	F02	A2		FS2	12/01/2004	11:23	39536
105	03/12/2003	15:32	653	3	WE06	03	F02	A3		FS3	26/01/2004	15:17	39578
106	23/12/2003	15:13	4	1	CCU					FS4	13/01/2004	16:35	39869
107	26/12/2003	15:05	653	1	CCU					FS4	04/01/2004	13:44	39889

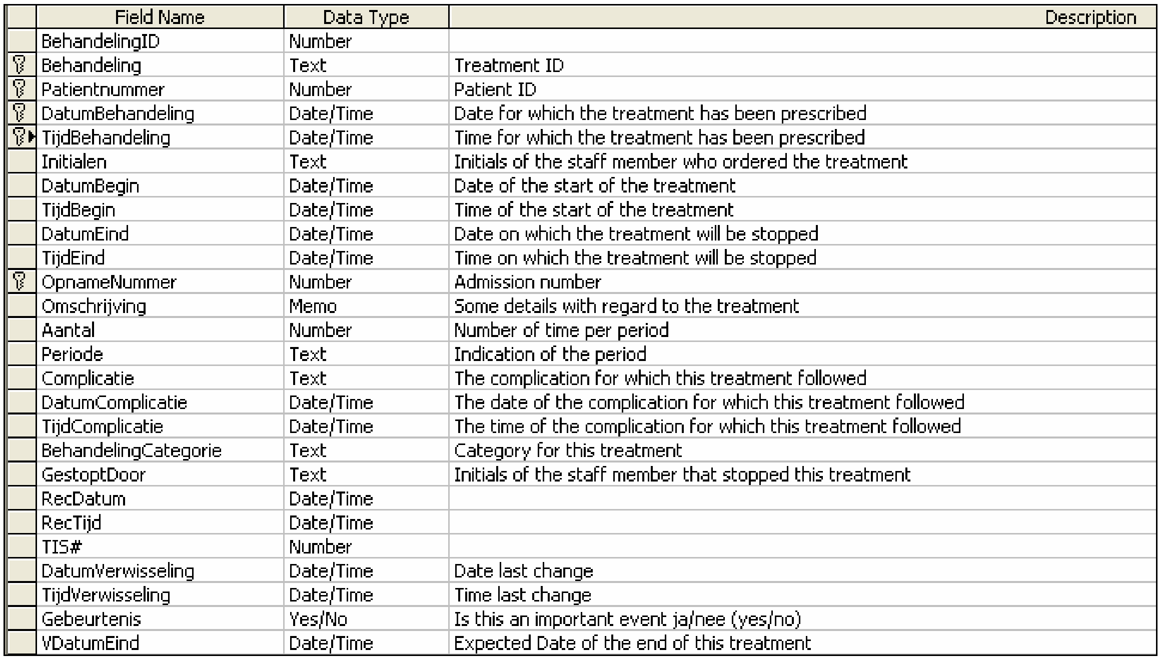
Opname	Spoed	Gestorven	Verantwoordelijk	VerantwoordelijkBegin	OverzichtDatum	OverzichtTijd	AIDS	Levercirrhose
Thuismedicatie:	True	Yes	INT		09/11/2003	14:30	No	No
Thuismedicatie: ASA; Omeprazol	True	No	INT	ANE	13/11/2003	15:07	No	No
LV Functie: goed, RVF goed, geen klepafwijkingen (TEE 01/12/03)	True	No	INT	ANE	30/11/2003	02:00	No	No
Thuismedicatie:	False	No	INT	ANE	01/12/2003	16:08	No	No
Thuismedicatie: monocedocard, adadlat, asa, atenolol, omnic, detrusitol	False	No	INT	ANE	03/12/2003	15:33	No	No
Thuismedicatie:	True	Yes	INT	ANE	23/12/2003	15:19	No	No
LV Functie: gedilateerd slecht	True	No	INT	ANE	26/12/2003	15:07	No	No

Nierinsuff	MaligniteitMeta	HematoOnco	HoofdDiagnose	HoofdDiagnoseCategorie	OntslagDiagnose	BSA	Heropname
No	No	No	Intensive Care	52 Respiratoir	Decompensatio Cordis	2.067161	Yes
No	No	No	Nazorg Long Chirurgie	02 Long-chirurgie	Lobectomie enkelzijdig	1.640893	No
No	No	No	Intensive Care	52 Respiratoir		2.069698	No
No	No	No	Nazorg Hartchirurgie	01 Cardio-chirurgie	CABG Meervoudig	1.730653	No
No	No	No	Nazorg Hartchirurgie	01 Cardio-chirurgie	Mediastinitis Re OK	2.052351	No
No	No	No	Intensive Care	52 Respiratoir	ARDS	1.495913	Yes
No	No	No	Intensive Care	51 Cardiovasculair	Respiratoire insuff.	2.032001	No

Figure B.24: Data in the Opname table

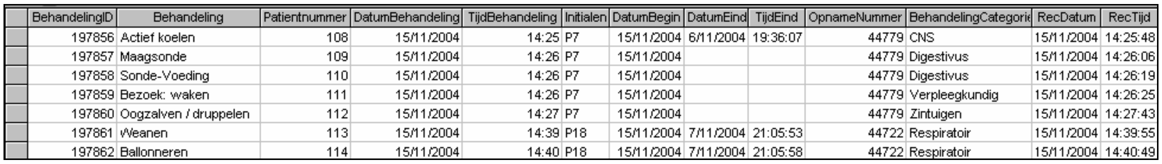
OpnameBehandeling

OpnameBehandeling table as the name suggests, stores information about a patient and his treatment. It makes use of the information given both in tables-*Opname* and *Behandeling*. It includes overview about the treatment given to him, duration of the treatment, treatment category, complication for which the treatment is being given etc. The total number of records in this table is 27740.



	Field Name	Data Type	Description
	BehandelingID	Number	
🔑	Behandeling	Text	Treatment ID
🔑	Patientnummer	Number	Patient ID
🔑	DatumBehandeling	Date/Time	Date for which the treatment has been prescribed
🔑▸	TijdBehandeling	Date/Time	Time for which the treatment has been prescribed
	Initialen	Text	Initials of the staff member who ordered the treatment
	DatumBegin	Date/Time	Date of the start of the treatment
	TijdBegin	Date/Time	Time of the start of the treatment
	DatumEind	Date/Time	Date on which the treatment will be stopped
	TijdEind	Date/Time	Time on which the treatment will be stopped
🔑	OpnameNummer	Number	Admission number
	Omschrijving	Memo	Some details with regard to the treatment
	Aantal	Number	Number of time per period
	Periode	Text	Indication of the period
	Complicatie	Text	The complication for which this treatment followed
	DatumComplicatie	Date/Time	The date of the complication for which this treatment followed
	TijdComplicatie	Date/Time	The time of the complication for which this treatment followed
	BehandelingCategorie	Text	Category for this treatment
	GestoptDoor	Text	Initials of the staff member that stopped this treatment
	RecDatum	Date/Time	
	RecTijd	Date/Time	
	TIS#	Number	
	DatumVerwisseling	Date/Time	Date last change
	TijdVerwisseling	Date/Time	Time last change
	Gebeurtenis	Yes/No	Is this an important event ja/nee (yes/no)
	VDatumEind	Date/Time	Expected Date of the end of this treatment

Figure B.25: Structure of the OpnameBehandeling table

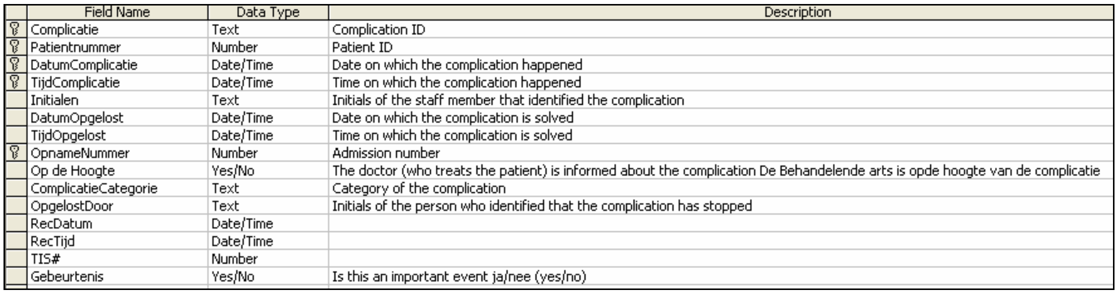


BehandelingID	Behandeling	Patientnummer	DatumBehandeling	TijdBehandeling	Initialen	DatumBegin	DatumEind	TijdEind	OpnameNummer	BehandelingCategorie	RecDatum	RecTijd
197856	Actief koelen	108	15/11/2004	14:25	P7	15/11/2004	6/11/2004	19:36:07	44779	CNS	15/11/2004	14:25:48
197857	Maagsonde	109	15/11/2004	14:26	P7	15/11/2004			44779	Digestivus	15/11/2004	14:26:06
197858	Sonde-Voeding	110	15/11/2004	14:26	P7	15/11/2004			44779	Digestivus	15/11/2004	14:26:19
197859	Bezoek: waken	111	15/11/2004	14:26	P7	15/11/2004			44779	Verpleegkundig	15/11/2004	14:26:25
197860	Oogzalven / druppelen	112	15/11/2004	14:27	P7	15/11/2004			44779	Zintuigen	15/11/2004	14:27:43
197861	Weanen	113	15/11/2004	14:39	P18	15/11/2004	7/11/2004	21:05:53	44722	Respiratoir	15/11/2004	14:39:55
197862	Ballonneren	114	15/11/2004	14:40	P18	15/11/2004	7/11/2004	21:05:58	44722	Respiratoir	15/11/2004	14:40:49

Figure B.26: Data in the OpnameBehandeling table

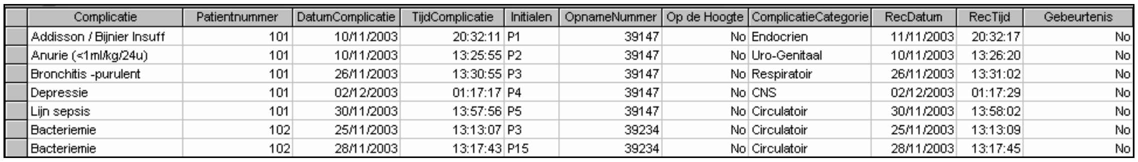
OpnameComplicatie

OpnameComplicatie table as the name suggests, stores information about a patient and the complication(s) from which he suffers. It includes overview about his complication, the time period when he suffers from this complication, the doctor/specialist responsible for his treatment etc. This table makes use of the information given both in tables-*Opname* and *Complicatie*. The total number of records in this table is 1518.



	Field Name	Data Type	Description
🔑	Complicatie	Text	Complication ID
🔑	Patientnummer	Number	Patient ID
🔑	DatumComplicatie	Date/Time	Date on which the complication happened
🔑	TijdComplicatie	Date/Time	Time on which the complication happened
	Initialen	Text	Initials of the staff member that identified the complication
	DatumOpgelost	Date/Time	Date on which the complication is solved
	TijdOpgelost	Date/Time	Time on which the complication is solved
🔑	OpnameNummer	Number	Admission number
	Op de Hoogte	Yes/No	The doctor (who treats the patient) is informed about the complication De Behandelende arts is opde hoogte van de complicatie
	ComplicatieCategorie	Text	Category of the complication
	OpgelostDoor	Text	Initials of the person who identified that the complication has stopped
	RecDatum	Date/Time	
	RecTijd	Date/Time	
	TIS#	Number	
	Gebeurtenis	Yes/No	Is this an important event ja/nee (yes/no)

Figure B.27: Structure of the OpnameComplicatie table

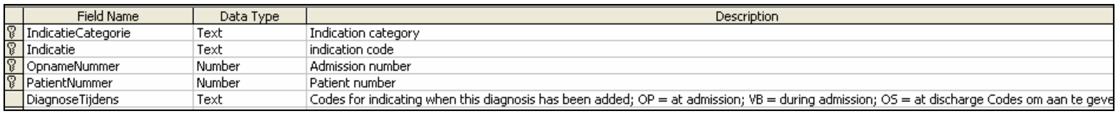


	Complicatie	Patientnummer	DatumComplicatie	TijdComplicatie	Initialen	OpnameNummer	Op de Hoogte	ComplicatieCategorie	RecDatum	RecTijd	Gebeurtenis
	Addisson / Bijnier Insuff	101	10/11/2003	20:32:11	P1	39147	No	Endocrien	11/11/2003	20:32:17	No
	Anurie (<1ml/kg/24u)	101	10/11/2003	13:25:55	P2	39147	No	Uro-Genitaal	10/11/2003	13:26:20	No
	Bronchitis -purulent	101	26/11/2003	13:30:55	P3	39147	No	Respiratoir	26/11/2003	13:31:02	No
	Depressie	101	02/12/2003	01:17:17	P4	39147	No	CNS	02/12/2003	01:17:29	No
	Lijn sepsis	101	30/11/2003	13:57:56	P5	39147	No	Circulatoir	30/11/2003	13:58:02	No
	Bacteriemie	102	25/11/2003	13:13:07	P3	39234	No	Circulatoir	25/11/2003	13:13:09	No
	Bacteriemie	102	28/11/2003	13:17:43	P15	39234	No	Circulatoir	28/11/2003	13:17:45	No

Figure B.28: Data in the OpnameComplicatie table

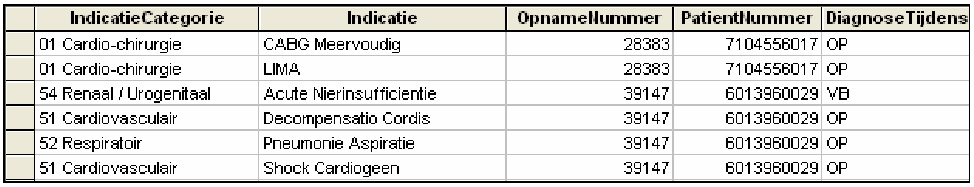
OpnameIndicatie

OpnameComplicatie table as the name suggests, stores information about a patient and the indication category for his complications. This table makes use of the information given both in tables-*Opname* and *Indicatie*. The total number of records in this table is 5519.



	Field Name	Data Type	Description
🔑	IndicatieCategorie	Text	Indication category
🔑	Indicatie	Text	indication code
🔑	OpnameNummer	Number	Admission number
🔑	PatientNummer	Number	Patient number
	DiagnoseTijdens	Text	Codes for indicating when this diagnosis has been added; OP = at admission; VB = during admission; OS = at discharge Codes om aan te geve

Figure B.29: Structure of the OpnameIndicatie table

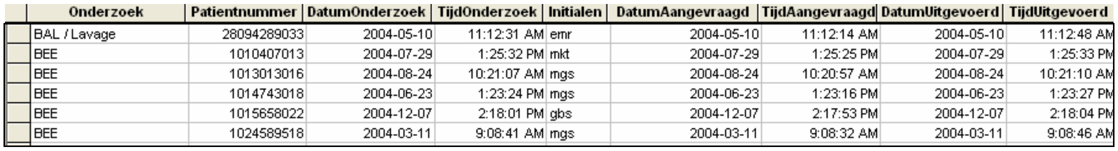


	IndicatieCategorie	Indicatie	OpnameNummer	PatientNummer	DiagnoseTijdens
	01 Cardio-chirurgie	CABG Meervoudig	28383	7104556017	OP
	01 Cardio-chirurgie	LIMA	28383	7104556017	OP
	54 Renaal / Urogenitaal	Acute Nierinsufficientie	39147	6013960029	VB
	51 Cardiovasculair	Decompensatio Cordis	39147	6013960029	OP
	52 Respiratoir	Pneumonie Aspiratie	39147	6013960029	OP
	51 Cardiovasculair	Shock Cardiogeen	39147	6013960029	OP

Figure B.30: Data in the OpnameIndicatie table

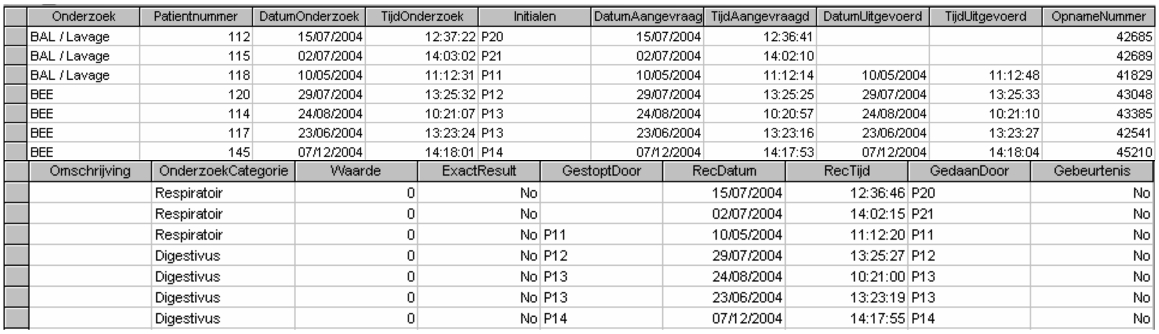
OpnameOnderzoek

OpnameOnderzoek table as the name suggests, stores information about a patient and the measuremnents/laboratory test(s) he undergoes. It includes information about the time and date for these tests, category of the tests, admission number of the patient, the specialist who performs the test etc. This table makes use of the information given both in tables-*Opname* and *Onderzoek*. The total number of records in this table is 9274.



	Onderzoek	Patientnummer	DatumOnderzoek	TijdOnderzoek	Initialen	DatumAangevraagd	TijdAangevraagd	DatumUitgevoerd	TijdUitgevoerd
	BAL / Lavage	28094289033	2004-05-10	11:12:31 AM	emr	2004-05-10	11:12:14 AM	2004-05-10	11:12:48 AM
	BEE	1010407013	2004-07-29	1:25:32 PM	mkt	2004-07-29	1:25:25 PM	2004-07-29	1:25:33 PM
	BEE	1013013016	2004-08-24	10:21:07 AM	mgs	2004-08-24	10:20:57 AM	2004-08-24	10:21:10 AM
	BEE	1014743018	2004-06-23	1:23:24 PM	mgs	2004-06-23	1:23:16 PM	2004-06-23	1:23:27 PM
	BEE	1015658022	2004-12-07	2:18:01 PM	gbs	2004-12-07	2:17:53 PM	2004-12-07	2:18:04 PM
	BEE	1024589518	2004-03-11	9:08:41 AM	mgs	2004-03-11	9:08:32 AM	2004-03-11	9:08:46 AM

Figure B.31: Structure of the OpnameOnderzoek table



	Onderzoek	Patientnummer	DatumOnderzoek	TijdOnderzoek	Initialen	DatumAangevraag	TijdAangevraagd	DatumUitgevoerd	TijdUitgevoerd	OpnameNummer
	BAL / Lavage	112	15/07/2004	12:37:22	P20	15/07/2004	12:36:41			42685
	BAL / Lavage	115	02/07/2004	14:03:02	P21	02/07/2004	14:02:10			42689
	BAL / Lavage	118	10/05/2004	11:12:31	P11	10/05/2004	11:12:14	10/05/2004	11:12:48	41829
	BEE	120	29/07/2004	13:25:32	P12	29/07/2004	13:25:25	29/07/2004	13:25:33	43048
	BEE	114	24/08/2004	10:21:07	P13	24/08/2004	10:20:57	24/08/2004	10:21:10	43385
	BEE	117	23/06/2004	13:23:24	P13	23/06/2004	13:23:16	23/06/2004	13:23:27	42541
	BEE	145	07/12/2004	14:18:01	P14	07/12/2004	14:17:53	07/12/2004	14:18:04	45210

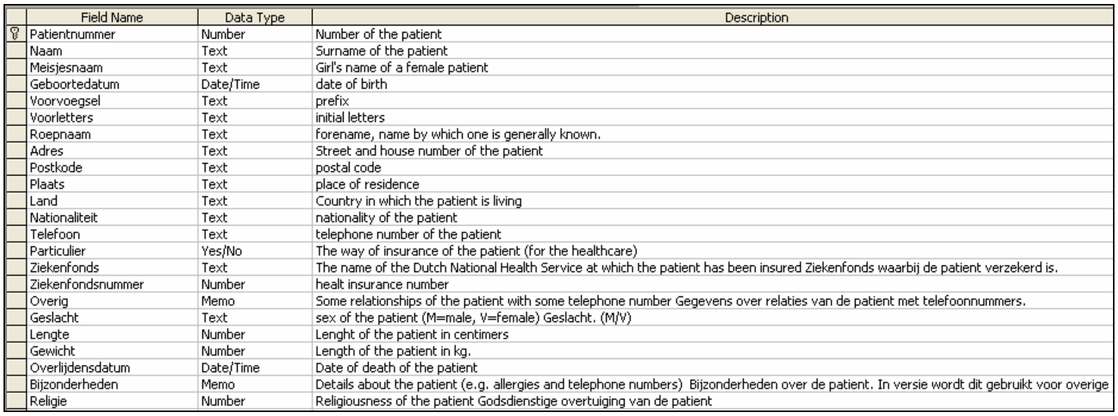
	Omschrijving	OnderzoekCategorie	Waarde	ExactResult	GestoptDoor	RecDatum	RecTijd	GedaanDoor	Gebeurtenis
		Respiratoir	0	No		15/07/2004	12:36:46	P20	No
		Respiratoir	0	No		02/07/2004	14:02:15	P21	No
		Respiratoir	0	No	P11	10/05/2004	11:12:20	P11	No
		Digestivus	0	No	P12	29/07/2004	13:25:27	P12	No
		Digestivus	0	No	P13	24/08/2004	10:21:00	P13	No
		Digestivus	0	No	P13	23/06/2004	13:23:19	P13	No
		Digestivus	0	No	P14	07/12/2004	14:17:55	P14	No

Figure B.32: Data in the OpnameOnderzoek table

Patient

Patient table as the name indicates, stores information about a patient. It stores personal and contact details of the patient, insurance details, nationality, religion his height, weight, patient number assigned to him etc.

The total number of records in this table is 23779. For reasons of confidentiality we do not show the data contained in this table.



	Field Name	Data Type	Description
🔑	Patientnummer	Number	Number of the patient
	Naam	Text	Surname of the patient
	Meisjesnaam	Text	Girl's name of a female patient
	Geboortedatum	Date/Time	date of birth
	Voorvoegsel	Text	prefix
	Voorletters	Text	initial letters
	Roepnaam	Text	forename, name by which one is generally known.
	Adres	Text	Street and house number of the patient
	Postkode	Text	postal code
	Plaats	Text	place of residence
	Land	Text	Country in which the patient is living
	Nationaliteit	Text	nationality of the patient
	Telefoon	Text	telephone number of the patient
	Particulier	Yes/No	The way of insurance of the patient (for the healthcare)
	Ziekenfonds	Text	The name of the Dutch National Health Service at which the patient has been insured Ziekenfonds waarbij de patient verzekerd is.
	Ziekenfondsnummer	Number	healt insurance number
	Overig	Memo	Some relationships of the patient with some telephone number Gegevens over relaties van de patient met telefoonnummers.
	Geslacht	Text	sex of the patient (M=male, V=female) Geslacht. (M/V)
	Lengte	Number	Lenght of the patient in centimers
	Gewicht	Number	Length of the patient in kg.
	Overlijdensdatum	Date/Time	Date of death of the patient
	Bijzonderheden	Memo	Details about the patient (e.g. allergies and telephone numbers) Bijzonderheden over de patient. In versie wordt dit gebruikt voor overige
	Religie	Number	Religiousness of the patient Godsdienstige overtuiging van de patient

Figure B.33: Structure of the Patient table

Besides these tables, we also mention some more tables present in the database. However, we do not show their structure and data contained in them for the already mentioned reasons. These are:

- ***Allergie***

This table stores information about allergies that a patient may suffer from. This table contains code for various allergies. The total number of records in this table is 18.

- ***Bedden***

This table stores information about bed and their corresponding rooms in the hospital. The total number of records in this table is 33.

- ***ComorbiditeitPeriode***

This table stores information about duration of a patient’s illness. The table stores this duration in terms of a fixed period. The total number of records in this table is 5.

- ***Eenheid***

This table indicates the measurement in which various medicines can be given, for example, mg, mg/capsule, % crème etc. The total number of records in this table is 65.

- ***Kamers***

This table stores information about the rooms in the hospital. The total number of records in this table is 18.

- ***Laboratorium***

This table stores information about various laboratory tests that can be done. It includes the type and name of the test, its cost price and other related information. The total number of records in this table is 466.

- ***OpnameAllergie***

OpnameAllergie table as the name suggests, stores information about a patient and the allergies he suffers from. It makes use of the information given both in tables-*Opname* and *Allergie*. The total number of records in this table is 466.

- ***OpnameBloedgas***

This table stores information about measurement (quantity) of various gases in a patient’s blood. The total number of records in this table is 28252.

- ***OpnameChemie***

This table stores information about measurement (quantity) of different chemicals in a patient's blood and body. The total number of records in this table is 19168.

- ***OpnameComorbiditeit***

This table stores information about how long a patient has been suffering from a particular complication. The total number of records in this table is 4693.

- ***OpnameDecubitus***

This table stores information about the mental, neurological state of a patient, his mobility state, nourishment status etc. The total number of records in this table is 824.

- ***Personeel***

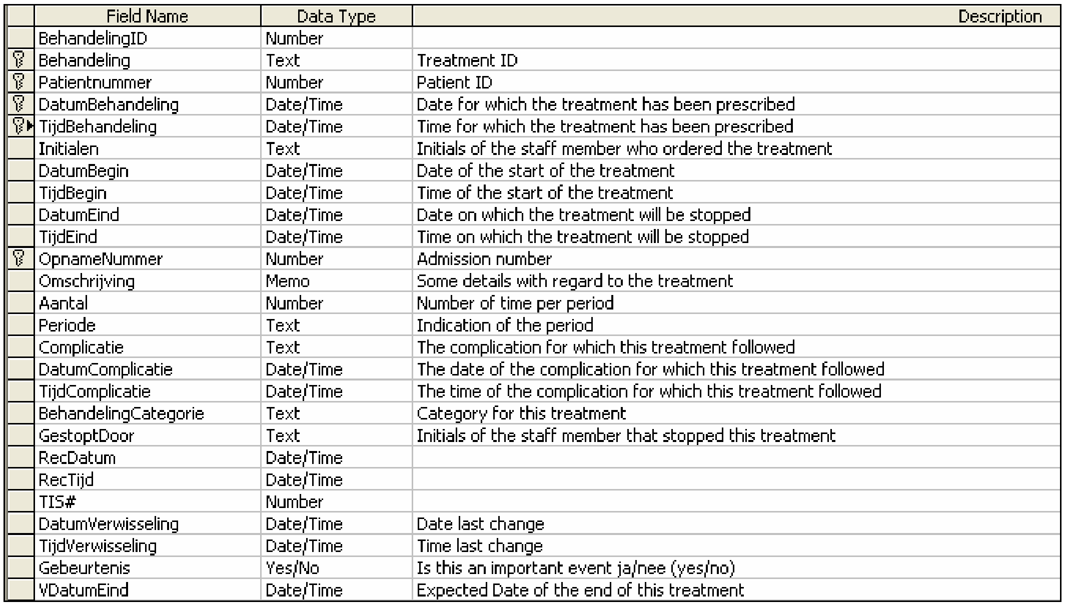
This table stores details about all the employees of the hospital. These employees may be nurses, doctors, and other supporting staff. The table includes their personal and contact details, their function in the hospital, their rights and permissions to access database etc. The total number of records in this table is 563.

C: A sample MS-Access table and its corresponding MXML log

For the purpose of understanding how does the data from an MS-Access table looks like when it is converted to the ProM's generic MXML format, here we take an example from Case study1. The table related to the treatment activities a patient undergoes, is selected. First we give the structure of the table i.e. what treatment information and in which format can be stored in this table. We then show some data contained in this table, which is later on converted to the MXML format.

Structure of the treatment table

Treatment table gives the details about treatment procedure for patients. It contains information about the patient in form of patient ID (the unique identifier for every patient), treatment prescribed to him, date and duration for which the treatment is prescribed to him, type of treatment and other treatment relevant information. The following screenshot (Figure C.1) shows the fields in this treatment (OpnameBehandling) table.



	Field Name	Data Type	Description
	BehandelingID	Number	
🔑	Behandeling	Text	Treatment ID
🔑	Patientnummer	Number	Patient ID
🔑	DatumBehandeling	Date/Time	Date for which the treatment has been prescribed
🔑▸	TijdBehandeling	Date/Time	Time for which the treatment has been prescribed
	Initialen	Text	Initials of the staff member who ordered the treatment
	DatumBegin	Date/Time	Date of the start of the treatment
	TijdBegin	Date/Time	Time of the start of the treatment
	DatumEind	Date/Time	Date on which the treatment will be stopped
	TijdEind	Date/Time	Time on which the treatment will be stopped
🔑	OpnameNummer	Number	Admission number
	Omschrijving	Memo	Some details with regard to the treatment
	Aantal	Number	Number of time per period
	Periode	Text	Indication of the period
	Complicatie	Text	The complication for which this treatment followed
	DatumComplicatie	Date/Time	The date of the complication for which this treatment followed
	TijdComplicatie	Date/Time	The time of the complication for which this treatment followed
	BehandelingCategorie	Text	Category for this treatment
	GestoptDoor	Text	Initials of the staff member that stopped this treatment
	RecDatum	Date/Time	
	RecTijd	Date/Time	
	TIS#	Number	
	DatumVerwisseling	Date/Time	Date last change
	TijdVerwisseling	Date/Time	Time last change
	Gebeurtenis	Yes/No	Is this an important event ja/nee (yes/no)
	VDatumEind	Date/Time	Expected Date of the end of this treatment

Figure C.1: Treatment table in Case study1

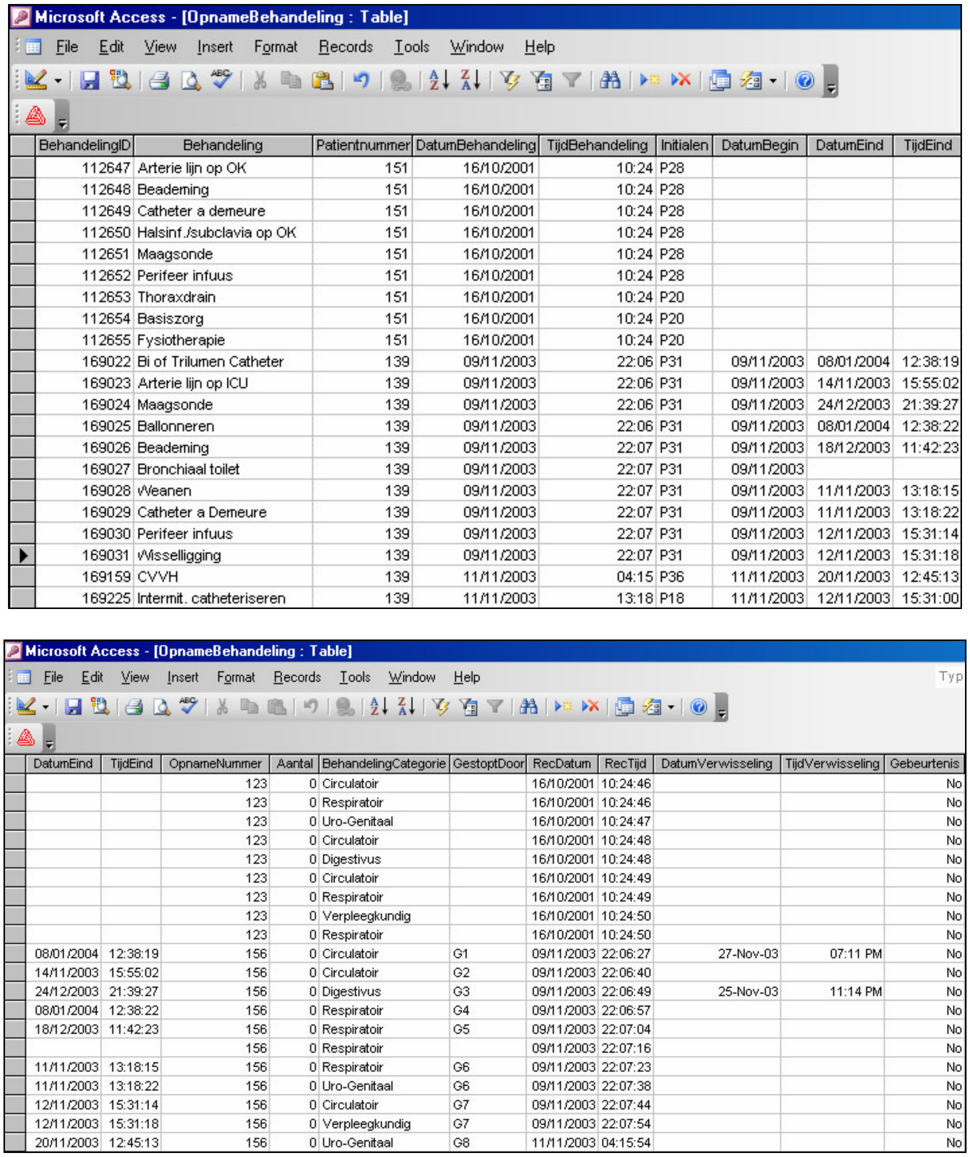
Data contained in the treatment table

Here we discuss the data contained in the treatment table of Figure C.1. Figure C.2a and C.2b shows data contained in this table.

Figure C.2 shows a portion of the data contained in this table:

- The column *OpnameNummer* identifies the patient's admission/visit to the hospital. Every time he visits the hospital he is given a new admission number but his unique *Patientnummer* remains the same.
- The treatment prescribed to each patient identified in *Patientnummer* is given in the column *Behandeling*. For example, the patient number 151 is given the treatment called Arterie lijn op OK, the date and time on which this treatment was prescribed was 16/10/2001, 10:24am and is recorded in the column *DatumBehandeling* and *TijdBehandeling*.
- The column *BehandelingCategorie* represents the category of the treatment which can be respiratory, circulatory, genitaal etc. For example, the patient 151 gets the treatment from various categories like circulatory, respiratory, digestive etc.

- Other information like the staff (physician) who prescribed and stopped the treatment, the complication for which the treatment is being given, date and time of complication etc. is also recorded in this table.


Microsoft Access - [OpnameBehandeling : Table]
File Edit View Insert Format Records Tools Window Help

BehandelingID	Behandeling	Patientnummer	DatumBehandeling	TijdBehandeling	Initialen	DatumBegin	DatumEind	TijdEind
112647	Arterie lijn op OK	151	16/10/2001	10:24	P28			
112648	Beademing	151	16/10/2001	10:24	P28			
112649	Catheter a demeure	151	16/10/2001	10:24	P28			
112650	Halsinf./subclavia op OK	151	16/10/2001	10:24	P28			
112651	Maagsonde	151	16/10/2001	10:24	P28			
112652	Perifeer infuus	151	16/10/2001	10:24	P28			
112653	Thoraxdrain	151	16/10/2001	10:24	P20			
112654	Basiszorg	151	16/10/2001	10:24	P20			
112655	Fysiotherapie	151	16/10/2001	10:24	P20			
169022	Bi of Trilumen Catheter	139	09/11/2003	22:06	P31	09/11/2003	08/01/2004	12:38:19
169023	Arterie lijn op ICU	139	09/11/2003	22:06	P31	09/11/2003	14/11/2003	15:55:02
169024	Maagsonde	139	09/11/2003	22:06	P31	09/11/2003	24/12/2003	21:39:27
169025	Ballonneren	139	09/11/2003	22:06	P31	09/11/2003	08/01/2004	12:38:22
169026	Beademing	139	09/11/2003	22:07	P31	09/11/2003	18/12/2003	11:42:23
169027	Bronchiaal toilet	139	09/11/2003	22:07	P31	09/11/2003		
169028	Weanen	139	09/11/2003	22:07	P31	09/11/2003	11/11/2003	13:18:15
169029	Catheter a Demeure	139	09/11/2003	22:07	P31	09/11/2003	11/11/2003	13:18:22
169030	Perifeer infuus	139	09/11/2003	22:07	P31	09/11/2003	12/11/2003	15:31:14
169031	Wisselligging	139	09/11/2003	22:07	P31	09/11/2003	12/11/2003	15:31:18
169159	CVVH	139	11/11/2003	04:15	P36	11/11/2003	20/11/2003	12:45:13
169225	Intermit. catheteriseren	139	11/11/2003	13:18	P18	11/11/2003	12/11/2003	15:31:00

Microsoft Access - [OpnameBehandeling : Table]
File Edit View Insert Format Records Tools Window Help
Typ

DatumEind	TijdEind	OpnameNummer	Aantal	BehandelingCategorie	GestoptDoor	RecDatum	RecTijd	DatumVerwisseling	TijdVerwisseling	Gebeurtenis
		123	0	Circulatoir		16/10/2001	10:24:46			No
		123	0	Respiratoir		16/10/2001	10:24:46			No
		123	0	Uro-Genitaal		16/10/2001	10:24:47			No
		123	0	Circulatoir		16/10/2001	10:24:48			No
		123	0	Digestivus		16/10/2001	10:24:48			No
		123	0	Circulatoir		16/10/2001	10:24:49			No
		123	0	Respiratoir		16/10/2001	10:24:49			No
		123	0	Verpleegkundig		16/10/2001	10:24:50			No
		123	0	Respiratoir		16/10/2001	10:24:50			No
08/01/2004	12:38:19	156	0	Circulatoir	G1	09/11/2003	22:06:27	27-Nov-03	07:11 PM	No
14/11/2003	15:55:02	156	0	Circulatoir	G2	09/11/2003	22:06:40			No
24/12/2003	21:39:27	156	0	Digestivus	G3	09/11/2003	22:06:49	25-Nov-03	11:14 PM	No
08/01/2004	12:38:22	156	0	Respiratoir	G4	09/11/2003	22:06:57			No
18/12/2003	11:42:23	156	0	Respiratoir	G5	09/11/2003	22:07:04			No
		156	0	Respiratoir		09/11/2003	22:07:16			No
11/11/2003	13:18:15	156	0	Respiratoir	G6	09/11/2003	22:07:23			No
11/11/2003	13:18:22	156	0	Uro-Genitaal	G6	09/11/2003	22:07:38			No
12/11/2003	15:31:14	156	0	Circulatoir	G7	09/11/2003	22:07:44			No
12/11/2003	15:31:18	156	0	Verpleegkundig	G7	09/11/2003	22:07:54			No
20/11/2003	12:45:13	156	0	Uro-Genitaal	G8	11/11/2003	04:15:54			No

Figure C.2Treatment related data

After taking a look at the data (Figure C.2) contained in the table for treatment of a patient, we now see the corresponding MXML log for this table. The MXML log for this table is obtained by following instructions in [34] and using the ProM Import framework. Figure C.3 shows a part of this MXML log. The information represented in the MXML log is consistent with the pre-defined DTD given in Appendix A.

- Process Instance ID: 123 (shown in Bold). PID is the *OpnameNummner* (admission number of a patient) in the database table.
- Patient Number: 151 (shown in Bold)
- We can see some data attributes for process instance. These includes Indiactie1, indicatieDiagnoseTijd etc.
- Each treatment activity is referred to as an AuditTrailEntry (ATE) in the log. This corresponds to the column *Behandeling* in Figure C.2.

- For each ATE information about its data attributes is also shown. These include the treatment category, TIS#, BelangrijkeGebeurtenis etc.
- We can check from the table in Figure C.2 that the patient number 151 undergoes nine treatments (*Behandeling*) and in the MXML log we have nine ATEs corresponding to each of these treatments.

```
<?xml version="1.0" encoding="UTF-8" ?>
<WorkflowLog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://is.tm.tue.nl/research/processmining/WorkflowLog.xsd" description="This log
is converted from the tables 'Process_Instances and Audit_Trail_Entries4 and Data_Attributes_Process_Instances and
Data_Attributes_Audit_Trail_Entries6' at the database 'jdbc:odbc:icisenglish'">
  <Data>
    <Attribute name="java.vendor">Sun Microsystems Inc.</Attribute>
    <Attribute name="java.version">1.5.0_06</Attribute>
    <Attribute name="mxml.version">1.0</Attribute>
    <Attribute name="os.arch">x86</Attribute>
    <Attribute name="os.name">Windows XP</Attribute>
    <Attribute name="os.version">5.1</Attribute>
    <Attribute name="user.name">s041914</Attribute>
  </Data>
  <Source program="MS Access database"/>
  <Process id="Process_Instances" description="A(n) MS Access database process.">
   <ProcessInstance id="123">
    <Data>
     <Attribute name="Comorbiditeit">Overige Locomotorius</Attribute>
     <Attribute name="ComorbiditeitCategorie">58 Huid /Subcutis /Spier/Bot</Attribute>
     <Attribute name="DuurOms"></Attribute>
     <Attribute name="Indicatie1">CABG Meervoudig</Attribute>
         <Attribute name="Indicatie2">LIMA</Attribute>
         <Attribute name="IndicatieCategorie1">01 Cardio-chirurgie</Attribute>
         <Attribute name="IndicatieCategorie2">01 Cardio-chirurgie</Attribute>
         <Attribute name="PatientNummer">151</Attribute>
         <Attribute name="indicatieDiagnoseTijd1">bij opname</Attribute>
         <Attribute name="indicatieDiagnoseTijd2">bij opname</Attribute>
       </Data>
      <AuditTrailEntry>
        <Data>
          <Attribute name="BelangrijkeGebeurtenis">0</Attribute>
          <Attribute name="ComplicatieCategorie">Circulatoir</Attribute>
          <Attribute name="Op_de_Hoogte">0</Attribute>
          <Attribute name="TIS#">0</Attribute>
          <Attribute name="typeTask">Complication</Attribute>
        </Data>
        <WorkflowModelElement>B_Arterie lijn op OK</WorkflowModelElement>
        <EventType>start</EventType>
        <Timestamp>2001-10-16T10:24:46.000+01:00</Timestamp>
        <Originator>P28</Originator>
      </AuditTrailEntry>
      <AuditTrailEntry>
        <Data>
          <Attribute name="BehandelingCategorie">Circulatoir</Attribute>
          <Attribute name="BelangrijkeGebeurtenis">0</Attribute>
          <Attribute name="TIS#">0</Attribute>
          <Attribute name="TypeTask">Treatment</Attribute>
        </Data>
        <WorkflowModelElement>B_Basiszorg</WorkflowModelElement>
        <EventType>start</EventType>
        <Timestamp>2001-10-16T10:24:50.000+01:00</Timestamp>
        <Originator>P28</Originator>
      </AuditTrailEntry>
      <AuditTrailEntry>
        <Data>
          <Attribute name="BehandelingCategorie">Verpleegkundig</Attribute>
          <Attribute name="BelangrijkeGebeurtenis">0</Attribute>
          <Attribute name="TIS#">0</Attribute>
          <Attribute name="TypeTask">Treatment</Attribute>
        </Data>
    <WorkflowModelElement>B_Beademing</WorkflowModelElement>
    <EventType>start</EventType>
    <Timestamp>2001-10-16T10:24:46.000+01:00</Timestamp>
    <Originator>P28</Originator>
  </AuditTrailEntry>
  <AuditTrailEntry>
    <Data>
      <Attribute name="BehandelingCategorie">Respiratoir</Attribute>
      <Attribute name="BelangrijkeGebeurtenis">0</Attribute>
      <Attribute name="TIS#">0</Attribute>
      <Attribute name="TypeTask">Treatment</Attribute>
    </Data>
    <WorkflowModelElement>B_Catheter a Demeure</WorkflowModelElement>
    <EventType>start</EventType>
    <Timestamp>2001-10-16T10:24:47.000+01:00</Timestamp>
    <Originator>P28</Originator>
  </AuditTrailEntry>
  <AuditTrailEntry>
    <Data>
      <Attribute name="BehandelingCategorie">Uro-Genitaal</Attribute>
      <Attribute name="BelangrijkeGebeurtenis">0</Attribute>
      <Attribute name="TIS#">0</Attribute>
      <Attribute name="TypeTask">Treatment</Attribute>
    </Data>
    <WorkflowModelElement>B_Fysiotherapie</WorkflowModelElement>
    <EventType>start</EventType>
    <Timestamp>2001-10-16T10:24:50.000+01:00</Timestamp>
    <Originator>P28</Originator>
  </AuditTrailEntry>
  <AuditTrailEntry>
    <Data>
      <Attribute name="BehandelingCategorie">Wond</Attribute>
      <Attribute name="BelangrijkeGebeurtenis">0</Attribute>
      <Attribute name="TIS#">0</Attribute>
      <Attribute name="TypeTask">Treatment</Attribute>
    </Data>
    <WorkflowModelElement>B_Halsinf./subclavia op Ok</WorkflowModelElement>
    <EventType>start</EventType>
    <Timestamp>2001-10-16T10:24:48.000+01:00</Timestamp>
    <Originator>P28</Originator>
  </AuditTrailEntry>
  <AuditTrailEntry>
    <Data>
      <Attribute name="BehandelingCategorie">Circulatoir</Attribute>
      <Attribute name="BelangrijkeGebeurtenis">0</Attribute>
      <Attribute name="TIS#">0</Attribute>
      <Attribute name="TypeTask">Treatment</Attribute>
    </Data>
    <WorkflowModelElement>B_Maagsonde</WorkflowModelElement>
    <EventType>start</EventType>
    <Timestamp>2001-10-16T10:24:48.000+01:00</Timestamp>
    <Originator>P28</Originator>
  </AuditTrailEntry>
  <AuditTrailEntry>
    <Data>
      <Attribute name="BehandelingCategorie">Respiratoir</Attribute>
      <Attribute name="BelangrijkeGebeurtenis">0</Attribute>
      <Attribute name="TIS#">0</Attribute>
      <Attribute name="TypeTask">Treatment</Attribute>
    </Data>
    <WorkflowModelElement>B_Perifeer infuus</WorkflowModelElement>
    <EventType>start</EventType>
    <Timestamp>2001-10-16T10:24:49.000+01:00</Timestamp>
    <Originator>P28</Originator>
  </AuditTrailEntry>
  <AuditTrailEntry>
    <Data>
      <Attribute name="BehandelingCategorie">Circulatoir</Attribute>
      <Attribute name="BelangrijkeGebeurtenis">0</Attribute>
      <Attribute name="TIS#">0</Attribute>
      <Attribute name="TypeTask">Treatment</Attribute>
    </Data>
    <WorkflowModelElement>B_Thoraxdrain</WorkflowModelElement>
    <EventType>start</EventType>
    <Timestamp>2001-10-16T10:24:49.000+01:00</Timestamp>
    <Originator>P28</Originator>
  </AuditTrailEntry>
</ProcessInstance>
```

Figure C.3: Corresponding MXML log

D: Parameters of the HeuristicsMiner

1. Positive Observations:

The concept behind the parameter: Positive Observations, available in the HM algorithm is explained. Consider the following figures to understand how to use this parameter:

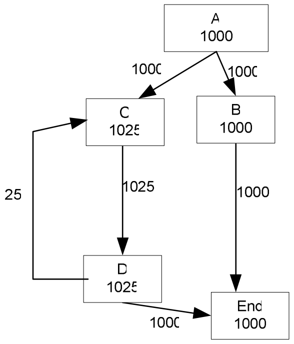


Figure D.1: Positive Observations=3

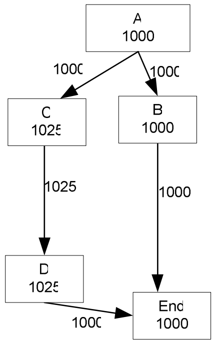


Figure D.2: Positive Observations=30

The value of positive observations threshold indicates the minimum required frequency of the relation a->b between any two activities a, b in the event log. As an example, in Figure D.1, Positive observations (PO) is set to value=3 and in figure D.2, PO is set to value=30. We can see that by setting a low value of positive observations we are also able to discover the relation D->C which is not visible when a higher value is set for positive observations. The discovery of the connection D->C when PO =3 shows that the task D is directly followed by task C at least three times or more. But when PO =30, this is not discovered as the task D may not be directly followed by task C at least 30 times. We can discover such low frequent dependencies if we set a low value to this parameter. This will help us to generate detailed behaviour. On the contrary, if we are interested only in the main behaviour of the event log we should set a high value for this parameter.

Thus, we see that positive observation threshold indicates that we will accept dependency relations between activities if the number of times one activity directly follows another is higher than or equal to the value of positive observations threshold. However, it is logical that the value cannot be set less than 1. If we try to give a value less than 1 in the plug-in it is automatically reset to the default value of 10.

2. Relative-to-best threshold:

Let us take some examples to understand the relative-to-best threshold:

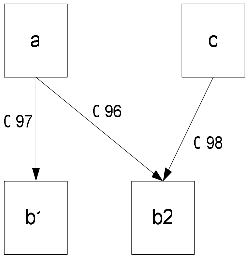


Figure D.3: Relative-to-best threshold=0.02

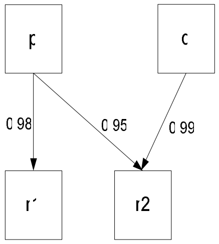


Figure D.4: Relative-to-best threshold=0.01

Example1:
Relative-to-best threshold =0.02
We need to find the dependency relations of initial activity 'a' with other activities. Considering Figure D.3 let us assume that the first dependency relation that we have obtained is a->b1 with a dependency value

=0.97. Suppose we also have connections- a->b2 and c->b2 and we have to decide whether the arc a->b2 would be present in the process model or not. (The arc c->b2 will be present in the process model because the activity b2 depends on the execution of activity c in the log. It is not related to any other activity in the log.) Now to decide about the arc a->b2, we calculate the difference between the dependency values of a->b1 and a->b2. We get the difference as (0.97-0.96) 0.01. The relative-to-best threshold is equal to 0.02. The difference 0.01 is lower than the value of relative-to-best threshold, hence the dependency relation a->b2 will be accepted and in the resulting process model we will have both the arcs- a->b2 and c->b2.

Example2:

Relative-to-best threshold =0.01
Considering Figure D.4, let us assume that the first dependency relation that we have obtained is p->r1 with a dependency value =0.98. Suppose we also have arcs- p->r2 and q->r2 and we have to now understand whether the arc p->r2 would be present in the process model or not. For this we calculate the difference between the dependency values of p->r1 and p->r2. We get the difference as (0.98-0.95) 0.03. The relative-to-best threshold is equal to 0.01. The difference 0.03 is greater than the value of relative-to-best threshold; hence this dependency relation p->r2 would not be accepted. And we will be left with the arcs p->r1 and q->r2 in our process model.

Suppose in this example we don't have the task q. We only have arcs p->r1 and p>r2. Now we have already seen that the arc p->r2 is not accepted because it fails to satisfy the relative-to-best threshold value. But since it satisfies the dependency threshold value (=0.9), it will be accepted in the process model iff the activity r2 has no other activity as its cause. Even if the parameter *all-activities-connected heuristic* is false, then also p->r2 would be accepted if the activity r2 has no other activity as its cause other than the activity p.

Example3:
Relative-to-best threshold = 0.05

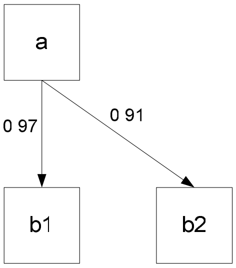


Figure D.5: Relative-to-best threshold=0.05

With reference to the Figure D.5, the difference of the dependency values of the two dependency relations a->b1 and a->b2 is calculated to be 0.06. This is greater than the value of relative-to-best threshold, and hence the connection a->b2 would not be accepted (a->b1 is the initial dependency relation). The value of relative-to-best threshold should be such that for the dependency relation a->b2 to be selected should be higher than 0.06. Thus, if the value of relative-to-best threshold is 0.07, the dependency relation a->b2 would be selected. So, we may conclude that a high value of relative-to-best threshold shall generate detailed behaviour as they would also include dependency relations with low dependency values.

It should be noted that the parameters *relative-to-best threshold, positive observations and dependency threshold* work in an AND relation to decide upon the dependency relations that should be present in the resulting process model. However, it should also be noted that the use of the parameter *all-activities-connected heuristic* overrides these parameters.

3. Extra Information

The extra information that the Heuristics Mining plug-in generates is shown below. The log used for generating Figure 3.8 in Section 3.3.1.9, Chapter 3, is taken as reference to explain this extra information.

1. Number of process instances, audit trail entries, number of connections and number of wrong observations.

2. Start information

```
00 5000 (a)
01 0000 (c)
02 0000 (g)
03 0000 (b)
04 0000 (e)
05 0000 (f)
06 0000 (i)
07 0000 (l)
08 0000 (n)
09 0000 (X)
10 0000 (d)
11 0000 (h)
12 0000 (m)
```

   As we can see it lists all the tasks in the log and indicates that the start place may contain 5000 tokens i.e. the process starts with 5000 cases. All other places initially have 0 tokens; this shows initially no activity is being executed.

3. End information:

```
00 0000 (a)
01 0000 (c)
02 0000 (g)
03 0000 (b)
04 0000 (e)
05 0000 (f)
06 0000 (i)
07 0000 (l)
08 0000 (n)
09 5000 (X)
10 0000 (d)
11 0000 (h)
12 0000 (m)
```

   As we can see it lists all the tasks in the log that were executed and indicates that the end place i.e. the task X contains 5000 tokens, this means that all the 5000 cases that entered the system were executed. All other places initially have 0 tokens indicating that no activity was pending i.e. none of the activity was left halfway during execution

4. Direct successors counters (|A > B|):
   This matrix gives the frequency of an activity A directly being followed by another activity B. The matrix gives this frequency for all the activities in the log. This information is given in the matrix below. It can be seen for example, that the activity *a* is directly followed by activity *c* 2525 times but not vice versa. It is true because in the process model in Figure 3.8 we can see that the activity *a* does not follow activity *c*. This way we can read the following matrix showing this information.

	00	01	02	03	04	05	06	07	08	09	10	11	12
00 a	0000	2525	0000	2475	0000	0000	0000	0000	0000	0000	0000	0000	0000
01 c	0000	0000	1532	1272	0288	0464	0000	0000	0000	0000	1444	0000	0000
02 g	0000	0000	0000	0942	0559	1002	1250	0000	0000	0000	0000	1247	0000
03 b	0000	1272	0465	0000	1485	1484	0000	0000	0000	0000	0294	0000	0000
04 e	0000	0281	0267	0000	0000	1790	0000	0000	0000	0000	0140	0000	0000
05 f	0000	0922	1004	0000	0000	0000	1218	0000	0000	0000	0571	1285	0000
06 i	0000	0000	0000	0000	0000	0000	0000	0432	1435	0000	0000	0000	0601
07 l	0000	0000	0000	0000	0000	0000	0000	0000	0360	0000	0000	0000	0072
08 n	0000	0000	0000	0000	0000	0000	0000	0000	0000	2468	0000	0000	0000
09 X	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
10 d	0000	0000	1732	0311	0146	0260	0000	0000	0000	0000	0000	0000	0000

11 h	0000	0000	0000	0000	0000	0000	0000	0000	0000	2532	0000	0000	0000
12 m	0000	0000	0000	0000	0000	0000	0000	0000	0673	0000	0000	0000	0000

5. Direct successors counters (|A>B>A|): This matrix gives the frequency of the length-two loop between activities A and B. This matrix is given in below. As seen there are no length-two-loops in the process model hence we have zero values in the matrix.

	00	01	02	03	04	05	06	07	08	09	10	11	12
00 a	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
01 c	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
02 g	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
03 b	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
04 e	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
05 f	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
06 i	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
07 l	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
08 n	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
09 X	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
10 d	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
11 h	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
12 m	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

6. All (also not accepted) L1L-dependency values: This matrix lists the frequency of one activity directly following itself. This matrix is given below. As there are no length-one-loops we have zero values in this matrix.

00	+0.000	(a)
01	+0.000	(c)
02	+0.000	(g)
03	+0.000	(b)
04	+0.000	(e)
05	+0.000	(f)
06	+0.000	(i)
07	+0.000	(l)
08	+0.000	(n)
09	+0.000	(X)
10	+0.000	(d)
11	+0.000	(h)
12	+0.000	(m)

7. All (also not accepted) L2L-dependency values: This matrix lists the frequency of one activity directly following another activity. This matrix also shows the rejected length-two-loop dependency relations.

	00	01	02	03	04	05	06	07	08	09	10	11	12
00 a	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
01 c	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
02 g	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
03 b	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
04 e	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
05 f	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
06 i	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
07 l	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
08 n	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
09 X	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
10 d	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
11 h	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
12 m	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000

8. All (also not accepted) A>B-dependency values: This matrix lists the dependency values of all those activities that directly follow some another activity. The matrix shows both the accepted and rejected dependency relations.

	00	01	02	03	04	05	06	07	08	09	10	11	12
00 a	+0.000	+1.000	+0.000	+1.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
01 c	-1.000	+0.000	+0.999	+0.000	+0.012	-0.330	+0.000	+0.000	+0.000	+0.000	+0.999	+0.000	+0.000
02 g	+0.000	-0.999	+0.000	+0.339	+0.353	-0.001	+0.999	+0.000	+0.000	+0.000	-0.999	+0.999	+0.000
03 b	-1.000	+0.000	-0.339	+0.000	+0.999	+0.999	+0.000	+0.000	+0.000	+0.000	-0.028	+0.000	+0.000
04 e	+0.000	-0.012	-0.353	-0.999	+0.000	+0.999	+0.000	+0.000	+0.000	+0.000	-0.021	+0.000	+0.000
05 f	+0.000	+0.330	+0.001	-0.999	-0.999	+0.000	+0.999	+0.000	+0.000	+0.000	+0.374	+0.999	+0.000
06 i	+0.000	+0.000	-0.999	+0.000	+0.000	-0.999	+0.000	+0.998	+0.999	+0.000	+0.000	+0.000	+0.998
07 l	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	-0.998	+0.000	+0.997	+0.000	+0.000	+0.000	+0.986
08 n	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	-0.999	-0.997	+0.000	+1.000	+0.000	+0.000	-0.999
09 X	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	-1.000	+0.000	+0.000	-1.000	+0.000
10 d	+0.000	-0.999	+0.999	+0.028	+0.021	-0.374	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
11 h	+0.000	+0.000	-0.999	+0.000	+0.000	-0.999	+0.000	+0.000	+0.000	+1.000	+0.000	+0.000	+0.000
12 m	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	-0.998	-0.986	+0.999	+0.000	+0.000	+0.000	+0.000

9. Accepted (L1L, L2L, A>B, A>>>B) dependency values: This matrix lists the accepted dependency values of all dependencies of the form L1, L2, A>B and A>>>B.

	00	01	02	03	04	05	06	07	08	09	10	11	12
00 a	+0.000	+1.000	+0.000	+1.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
01 c	+0.000	+0.000	+0.999	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.999	+0.000	+0.000
02 g	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.999	+0.000	+0.000	+0.000	+0.000	+0.999	+0.000
03 b	+0.000	+0.000	+0.000	+0.000	+0.999	+0.999	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
04 e	+0.000	+0.000	+0.000	+0.000	+0.000	+0.999	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
05 f	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.999	+0.000	+0.000	+0.000	+0.000	+0.999	+0.000
06 i	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.998	+0.999	+0.000	+0.000	+0.000	+0.998
07 l	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.997	+0.000	+0.000	+0.000	+0.986
08 n	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+1.000	+0.000	+0.000	+0.000
09 X	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
10 d	+0.000	+0.000	+0.999	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
11 h	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+1.000	+0.000	+0.000	+0.000
12 m	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.999	+0.000	+0.000	+0.000	+0.000

10. All IN and-values.

	00	01	02	03	04	05	06	07	08	09	10	11	12
00 a	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
01 c	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
02 g	+0.000	+0.000	+0.000	+0.000	+0.000	+0.792	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
03 b	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
04 e	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
05 f	+0.000	+0.000	+0.792	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
06 i	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
07 l	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
08 n	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
09 X	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
10 d	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
11 h	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
12 m	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000

11. All OUT and-values.

	00	01	02	03	04	05	06	07	08	09	10	11	12
00 a	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
01 c	+0.000	+0.000	+0.000	+0.509	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
02 g	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
03 b	+0.000	+0.509	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
04 e	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
05 f	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
06 i	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
07 l	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
08 n	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
09 X	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
10 d	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
11 h	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000
12 m	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000

12. Total number of connections: It gives the number of total arcs in the dependency graph. In this process model we have 20 arcs connecting different activities.

13. "Wrong" observations (#B>A but A->B accepted) between accepted dependency relations: This indicates the wrong connections which have been accepted for generating the process model. But in this case the zero values in the following matrix indicate that the number of wrong observations is zero.

	00	01	02	03	04	05	06	07	08	09	10	11	12
00 a	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
01 c	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
02 g	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
03 b	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
04 e	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
05 f	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
06 i	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
07 l	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
08 n	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
09 X	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
10 d	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
11 h	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
12 m	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

E: MXML logs from Case study1

In this appendix, the MXML log fragments for Case study1 used for experiments with different algorithms, are given. Logs with information about complications and treatments were used in different forms, either the original log was used or these logs were filtered for specific complication categories like *CNS* etc. Only the original logs are presented here because other logs are derived from them. For each log, we represent one process instance including its data attributes, audit trail entries, and other information. The structure of the log is consistent with the pre-defined DTD given in Appendix A.

1. Complications log

To obtain the PI information the following tables are combined:

- *Patient*: This table gives the general information about any patient
- *Opname*: Details about the patients admission are obtained from this table
- *OpnameIndicatieEnDiagnoseTijd*: This table gives details about indications about any patient's complications and the diagnosis time when information about indications was recorded.
- *ComorbiditeitEnDuur*: This table gives information about the duration from which a patient is suffering from some complication(s).

For every patient the information obtained from these tables is recorded. To obtain complication-specific information for the patients, the table *OpnameComplicatie* is used. All these tables were combined to construct the log for complications. After conversion into MXML, the following log (only a part of log is shown here) is obtained:

```
<?xml version="1.0" encoding="UTF-8"?>
<WorkflowLog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="WorkflowLog.xsd" description="Exported by ProM framework from This log is
converted from the tables 'Process_Instances and Audit_Trail_Entries3 and Data_Attributes_Process_Instances and
Data_Attributes_Audit_Trail_Entries3' at the database 'jdbc:odbc:icisenglish'">
<Data>
<Attribute name="mxml.version">1.0</Attribute>
<Attribute name="java.version">1.5.0_06</Attribute>
<Attribute name="user.name">s041914</Attribute>
<Attribute name="os.name">Windows XP</Attribute>
<Attribute name="os.arch">x86</Attribute>
<Attribute name="java.vendor">Sun Microsystems Inc.</Attribute>
<Attribute name="os.version">5.1</Attribute>
</Data>
<Source program="MS Access database">
<Data>
<Attribute name="program">MS Access database</Attribute>
</Data>
</Source>
<Process id="Process_Instances" description="A(n) MS Access database process.">
<ProcessInstance id="124" description="">
<Data>
<Attribute name="HoofdDiagnose">Intensive Care</Attribute>
<Attribute name="ComorbiditeitCategorie3">51 Cardiovasculair</Attribute>
<Attribute name="DuurOms3"></Attribute>
<Attribute name="indicatieDiagnoseTijd4">bij opname</Attribute>
<Attribute name="VerantwoordelijkBegin"></Attribute>
<Attribute name="Gestorven">True</Attribute>
<Attribute name="Comorbiditeit1">CABG Meervoudig</Attribute>
<Attribute name="DuurOms2">&lt;3 mnd</Attribute>
<Attribute name="MaligniteitMeta">False</Attribute>
<Attribute name="DuurOms1">&lt;3 mnd</Attribute>
<Attribute name="indicatieDiagnoseTijd2">bij opname</Attribute>
<Attribute name="OpnameTimestamp">09-11-2003 14:29:27</Attribute>
<Attribute name="indicatieDiagnoseTijd1">bij verblijf</Attribute>
<Attribute name="RetourAfdeling">MORT</Attribute>
<Attribute name="Bednummer">3</Attribute>
<Attribute name="Indicatie1">Acute Nierinsufficientie</Attribute>
<Attribute name="ComorbiditeitCategorie2">06 Urologische chirurgie</Attribute>
<Attribute name="PatientNummer2">101</Attribute>
```

```
<Attribute name="Verantwoordelijk">INT</Attribute>
<Attribute name="AIDS">False</Attribute>
<Attribute name="HoofdDiagnoseCategorie">52 Respiratoir</Attribute>
<Attribute name="Kamernummer">653</Attribute>
<Attribute name="Patientnummer">101</Attribute>
<Attribute name="OntslagDiagnose">Decompensatio Cordis</Attribute>
<Attribute name="IndicatieCategorie2">51 Cardiovasculair</Attribute>
<Attribute name="Indicatie2">Decompensatio Cordis</Attribute>
<Attribute name="IndicatieCategorie3">52 Respiratoir</Attribute>
<Attribute name="age_patient_admission">64</Attribute>
<Attribute name="PatientNummer1">101</Attribute>
<Attribute name="Afdelingscode">CCU</Attribute>
<Attribute name="Comorbiditeit3">Hartfalen Chronisch</Attribute>
<Attribute name="Heropname">True</Attribute>
<Attribute name="OntslagTimestamp">08-01-2004 22:14:25</Attribute>
<Attribute name="ComorbiditeitCategorie1">01 Cardio-chirurgie</Attribute>
<Attribute name="HematoOnco">False</Attribute>
<Attribute name="Gewicht">92</Attribute>
<Attribute name="Toestand_Bij_Ontslag">8</Attribute>
<Attribute name="Spoed">True</Attribute>
<Attribute name="Indicatie4">Shock Cardiogeen</Attribute>
<Attribute name="indicatieDiagnoseTijd3">bij opname</Attribute>
<Attribute name="IndicatieCategorie4">51 Cardiovasculair</Attribute>
<Attribute name="IndicatieCategorie1">54 Renaal / Urogenitaal</Attribute>
<Attribute name="Geslacht">M</Attribute>
<Attribute name="PatientNummer3">101</Attribute>
<Attribute name="Nierinsuff">False</Attribute>
<Attribute name="Levercirrhose">False</Attribute>
<Attribute name="Toestand_Bij_Opname">1</Attribute>
<Attribute name="Comorbiditeit2">Nefrectomie</Attribute>
<Attribute name="Lengte">174</Attribute>
<Attribute name="COMA">False</Attribute>
<Attribute name="Indicatie3">Pneumonie Aspiratie</Attribute>
<Attribute name="Body_Surface_Area_in_m2">2,067161</Attribute>
</Data>
<AuditTrailEntry>
<WorkflowModelElement>ArtificialStartTask</WorkflowModelElement>
<EventType>complete</EventType>
<Originator>Artificial (ProM)</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="ComplicatieCategorie">Uro-Genitaal</Attribute>
<Attribute name="typeTask">Complication</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="Op_de_Hoogte">0</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>C_Anurie (&lt;1ml/kg/24u)</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2003-11-10T13:25:55.000+01:00</Timestamp>
<Originator>P2</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="ComplicatieCategorie">Endocrien</Attribute>
<Attribute name="typeTask">Complication</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="Op_de_Hoogte">0</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>C_Addisson / Bijnier Insuff</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2003-11-10T20:32:11.000+01:00</Timestamp>
<Originator>P1</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="ComplicatieCategorie">Respiratoir</Attribute>
<Attribute name="typeTask">Complication</Attribute>
```

```
<Attribute name="TIS#">0</Attribute>
<Attribute name="Op_de_Hoogte">0</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>C_Bronchitis -purulent</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2003-11-26T13:30:55.000+01:00</Timestamp>
<Originator>P3</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="ComplicatieCategorie">Circulatoir</Attribute>
<Attribute name="typeTask">Complication</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="Op_de_Hoogte">0</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>C_Lijn sepsis</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2003-11-30T13:57:56.000+01:00</Timestamp>
<Originator>P5</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="ComplicatieCategorie">CNS</Attribute>
<Attribute name="typeTask">Complication</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="Op_de_Hoogte">0</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>C_Depressie</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2003-12-02T01:17:17.000+01:00</Timestamp>
<Originator>P4</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<WorkflowModelElement>ArtificialEndTask</WorkflowModelElement>
<EventType>complete</EventType>
<Originator>Artificial (ProM)</Originator>
</AuditTrailEntry>
</ProcessInstance>
```

Figure E.1: MXML log fragment for the *Complications* log

2. Treatments_log

To obtain the PI information the following tables are combined:

- *OpnameIndicatieEnDiagnoseTijd*: This table gives details about indications about any patient's complications and the diagnosis time when information about indications was recorded.
- *ComorbiditeitEnDuur*: This table gives information about the duration from which a patient is suffering from some complication(s).

Information obtained from these tables is recorded for every patient. To obtain treatment-specific information for the patients, the table *OpnameBehandeling* is used. All these tables were combined to construct the log for complications. After conversion into MXML, the following log (only a part of log is shown here) is obtained:

```
<?xml version="1.0" encoding="UTF-8"?>
<WorkflowLog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="WorkflowLog.xsd" description="Exported by ProM framework from This log is converted
from the tables 'Process_Instances and Audit_Trail_Entries4 and Data_Attributes_Process_Instances and
Data_Attributes_Audit_Trail_Entries6' at the database 'jdbc:odbc:icisenglish'">
<Data>
<Attribute name="mxml.version">1.0</Attribute>
<Attribute name="java.version">1.5.0_06</Attribute>
<Attribute name="user.name">s041914</Attribute>
<Attribute name="os.name">Windows XP</Attribute>
```

```
<Attribute name="os.arch">x86</Attribute>
<Attribute name="java.vendor">Sun Microsystems Inc.</Attribute>
<Attribute name="os.version">5.1</Attribute>
</Data>
<Source program="MS Access database">
<Data>
<Attribute name="program">MS Access database</Attribute>
</Data>
</Source>
<Process id="Process_Instances" description="A(n) MS Access database process.">
<ProcessInstance id="123" description="">
<Data>
<Attribute name="IndicatieCategorie1">01 Cardio-chirurgie</Attribute>
<Attribute name="PatientNummer">151</Attribute>
<Attribute name="Indicatie1">CABG Meervoudig</Attribute>
<Attribute name="ComorbiditeitCategorie">58 Huid /Subcutis /Spier /Bot</Attribute>
<Attribute name="indicatieDiagnoseTijd2">bij opname</Attribute>
<Attribute name="Comorbiditeit">Overige Locomotorius</Attribute>
<Attribute name="DuurOms"></Attribute>
<Attribute name="IndicatieCategorie2">01 Cardio-chirurgie</Attribute>
<Attribute name="indicatieDiagnoseTijd1">bij opname</Attribute>
<Attribute name="Indicatie2">LIMA</Attribute>
</Data>
<AuditTrailEntry>
<WorkflowModelElement>ArtificialStartTask</WorkflowModelElement>
<EventType>complete</EventType>
<Originator>Artificial (ProM)</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="ComplicatieCategorie">Circulatoir</Attribute>
<Attribute name="typeTask">Complication</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="Op_de_Hoogte">0</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>B_Arterie lijn op OK</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2001-10-16T11:24:46.000+02:00</Timestamp>
<Originator>P31</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="TypeTask">Treatment</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="BehandelingCategorie">Verpleegkundig</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>B_Beademing</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2001-10-16T11:24:46.000+02:00</Timestamp>
<Originator>P31</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="TypeTask">Treatment</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="BehandelingCategorie">Respiratoir</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>B_Catheter a Demeure</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2001-10-16T11:24:47.000+02:00</Timestamp>
<Originator>P31</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="TypeTask">Treatment</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="BehandelingCategorie">Wond</Attribute>
```

```
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>B_Halsinf./subclavia op Ok</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2001-10-16T11:24:48.000+02:00</Timestamp>
<Originator>P31</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="TypeTask">Treatment</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="BehandelingCategorie">Circulatoir</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>B_Maagsonde</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2001-10-16T11:24:48.000+02:00</Timestamp>
<Originator>P31</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="TypeTask">Treatment</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="BehandelingCategorie">Respiratoir</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>B_Perifeer infuus</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2001-10-16T11:24:49.000+02:00</Timestamp>
<Originator>P31</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="TypeTask">Treatment</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="BehandelingCategorie">Circulatoir</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>B_Thoraxdrain</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2001-10-16T11:24:49.000+02:00</Timestamp>
<Originator>P31</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="TypeTask">Treatment</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="BehandelingCategorie">Circulatoir</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>B_Basiszorg</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2001-10-16T11:24:50.000+02:00</Timestamp>
<Originator>P31</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="TypeTask">Treatment</Attribute>
<Attribute name="TIS#">0</Attribute>
<Attribute name="BehandelingCategorie">Uro-Genitaal</Attribute>
<Attribute name="BelangrijkeGebeurtenis">0</Attribute>
</Data>
<WorkflowModelElement>B_Fysiotherapie</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2001-10-16T11:24:50.000+02:00</Timestamp>
<Originator>P31</Originator>
</AuditTrailEntry>
<AuditTrailEntry>
<WorkflowModelElement>ArtificialEndTask</WorkflowModelElement>
<EventType>complete</EventType>
```

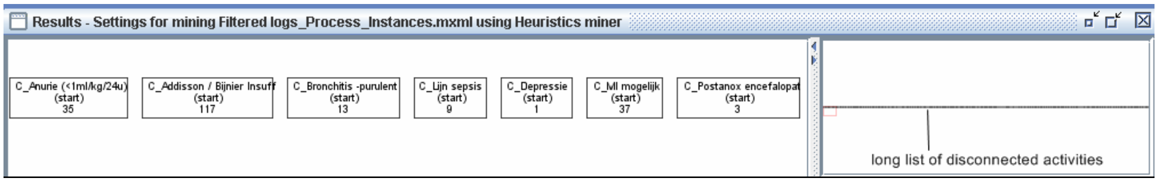
```
<Originator>Artificial (ProM)</Originator>
</AuditTrailEntry>
</ProcessInstance>
```

Figure E.2: MXML log fragment for the *Treatments* log

F: Effect of the *all-activities-connected heuristic* HM parameter

In this appendix, the effect of the HM parameter *all-activities-connected heuristic* is illustrated. As already mentioned in Chapter 3, Section 3.3.1.1, the use of this parameter overrides all other parameters of the algorithm. To see the impact of not choosing this parameter, some experiments were performed on the complications and treatments logs from the Case study1. This is discussed below:

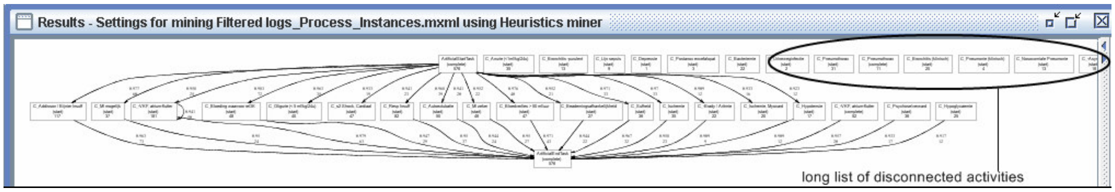
For the **complications log** used in Chapter 3, Section 3.4.1, Illustration 1 when the parameter *all-activities-connected heuristic* is not selected and heuristics mining is carried out with the default parameter values, the resulting dependency graph is as shown in Figure F.1. It should be noted that the log does not have unique start and end points.



Results - Settings for mining Filtered logs_Process_Instances.mxml using Heuristics miner

Figure F.1: Dependency graph for complications log, *all-activities-connected heuristic=false*

It is seen that all the 185 activities of the log are disconnected. If unique start and end points are added to this log, we obtain the dependency graph as shown in Figure F.2. In this figure, though a simpler process model is seen, the model is not free from dangling activities and missing connections. The non-dangling activities are connected to only the start and end points, and there is no interconnection between these activities. Besides, it is also observed that most of the low frequent events are totally disconnected from the rest of the model.



Results - Settings for mining Filtered logs_Process_Instances.mxml using Heuristics miner

Figure F.2: Dependency graph for complications log with unique start and end points, *all-activities-connected heuristic=false*

For the **treatments log** used in Chapter 3, Section 3.4.1, Illustration 1 when *all-activities-connected heuristic* parameter is not selected and mining is carried out with the default parameter values, the resulting dependency graph is as shown in Figure F.3. It should be noted that the log does not have unique start and end points. The model is undoubtedly better than the spaghetti model in Figure 3.12 and captures the main behaviour of the log. The low frequent behaviour is seen as dangling activities. If unique start and end points are added to this log, a dependency graph as shown in Figure F.4 is obtained. It is also observed that some not so low frequent activities are also not on the path from the start to the end point.

From the various experiments conducted to study the impact of this parameter on the output of the HM algorithm, it is found that when this parameter is false then for some logs, the mined model is easier to understand but it contains a lot of disconnected activities. For some logs, like the complications log, we only obtain a list of disconnected activities and nothing else. This is so because when this parameter is used, other parameters of the algorithm are ignored and a dependency graph is generated on its basis. For Case study 2 also, when *all-activities connected heuristic* is false, only three activities are found to be connected to each other and rest all are totally disconnected. And at the same time, it is also found that when this parameter is true, the model obtained is very complex and confusing. Therefore, it could not be concluded whether it is better to use the *all-activities connected heuristic* or not.

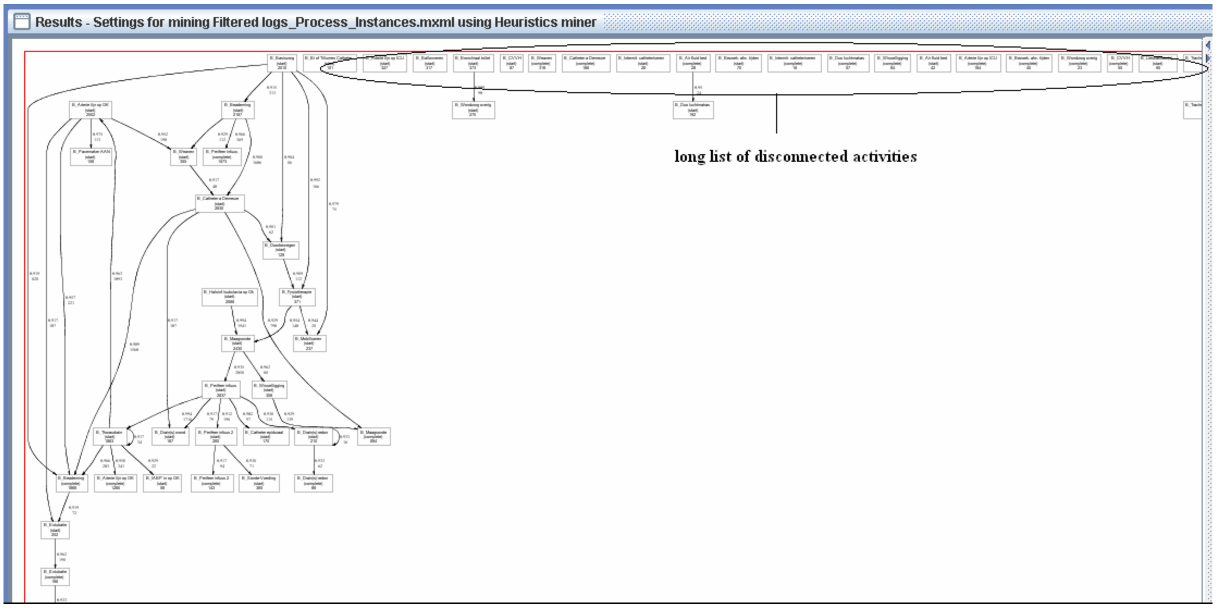

Results - Settings for mining Filtered logs_Process_Instances.mxml using Heuristics miner
long list of disconnected activities

Figure F.3: Dependency graph for treatments log, *all-activities-connected heuristic=false*

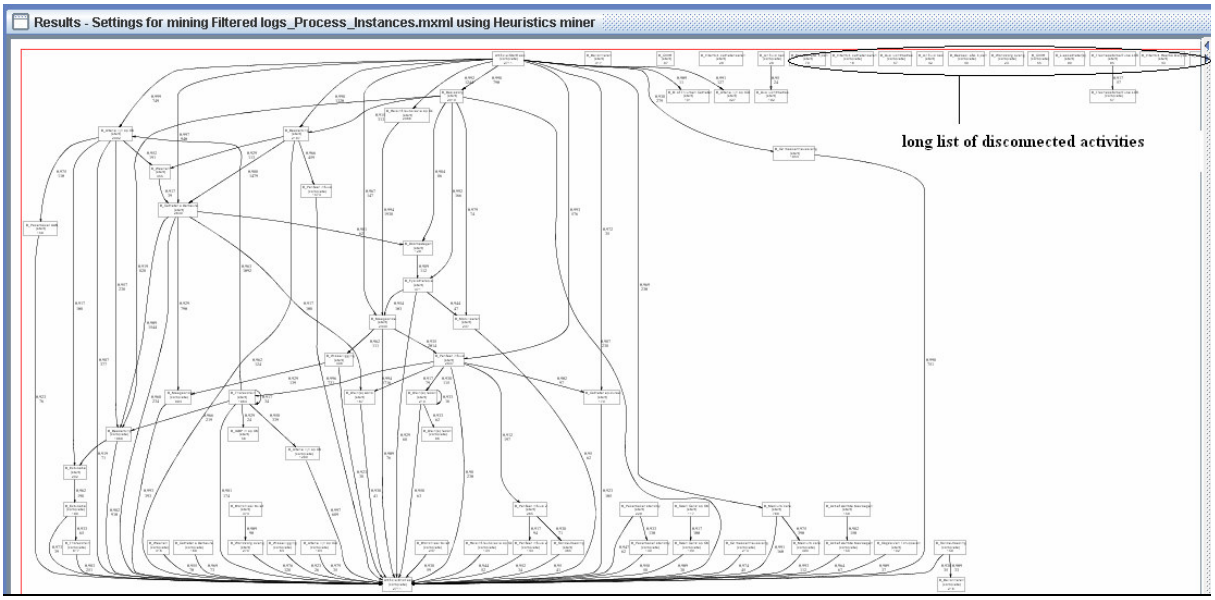

Results - Settings for mining Filtered logs_Process_Instances.mxml using Heuristics miner
long list of disconnected activities

Figure F.4: Dependency graph for treatments log with unique start and end points, *all-activities-connected heuristic=false*

G: Fitness of process models

The outcome of process mining techniques focusing on the control-flow perspective is a process model underlying the input event log. One of the issues concerning this mining effort is that whether the event log and the generated process model conform to each other. The misalignment between the event log and the mined process model may reflect that the reality is not as it is expected to be. Further measures can be then taken to bridge the gap between the perceived reality and the actual reality. This misalignment is discussed in [24] as conformance testing.

Conformance testing or conformance analysis, aims at the detection of inconsistencies between a process model and its corresponding execution log, and the quantification of the gap. To make this operational one needs to define metrics. This paper also introduces two dimensions of conformance. The first dimension is fitness, which can be characterized by the question "Does the observed process comply with the control flow specified by the process model?". The second is appropriateness, which can be associated with the question "Does the model describe the observed process in a suitable way?". The HM algorithm makes use of one of these conformance dimensions to give information about the mined process model.

The HM provides the value of the fitness of the mined process model to indicate if the log conforms to the mined process model. Generally simple fitness measure implies the number of correctly parsed traces over the total number of traces in the log. But this simple fitness measure is too naive as it gives a very coarse indication about a process model's compliance to a given log. There is another fitness measure known as *stop semantics fitness measure*. This fitness measure value is the result of replaying the log and stopping the parsing process whenever a parsing error occurs. But this is also not a good measure because we may not be able to differentiate if the parsing stopped in an early stage or towards the completion stage. Therefore, we have yet another measure of fitness, known as *Continuous semantics fitness*. In this procedure the parsing does not stop after identification of an error. Instead the error is registered and the parsing is continued. The HM gives us this fitness value. *Improved continuous fitness measure* is also given by the HM.

H: Completeness and Soundness in the DWS Approach

The Disjunctive workflow schema approach iteratively generates workflow schema, and this is refinement is guided by notions of completeness and soundness. Here we give a brief overview of these notions. Readers are directed to [11] for further reading.

The aim of process mining techniques is to analyze the event log generated of an information system running in a company and to identify the process model encompassing the various activities of that company. But many such models can be mined and therefore we need a model which is the most conformant, i.e. the model which is the best aligned with the log. One of the popular and widely used conformance measures is the *completeness or fitness* of the process model. This means we look for the percentage of log traces that may be the result of some enactment supported by the mined model. But in this case such a complete model may also support some extra behavior that are not present in the event log but are possible. So, we need to focus on models which are not so generic in nature and depict only the appropriate behavior. This is captured by the *soundness* measure. It is also referred to as minimality or behavioral appropriateness. Soundness measures the percentage of enactments of the mined model that are also registered in the log. So, the lower the value of soundness more is the extra behavior depicted by the mined process model.

In context of disjunctive workflow schema, below we give the formulae representing the notions of completeness and soundness [11]. The DWS approach aims at discovering a disjunctive workflow schema $WS^{\vee}$ for a process P which is α-sound and β-complete, for some given alpha and beta. Soundness measure is represented by the variable α and the variable β represents the notion of completeness.

Soundness is the percentage of traces compliant with $WS^{\vee}$ that have been registered in the log. The larger is the value of soundness the sounder is the process model. It is given as:

$$Soundness(WS^{\vee}, L_p) = \left(\frac{|\{s \mid s \in L_p \wedge s \models WS^{\vee}\}|}{|\{s \mid s \models WS^{\vee}\}|} \right)$$

Equation H.1: Soundness

Completeness is the percentage of traces in the log that are compliant with $WS^{\vee}$. The larger the value the more complete is the process model. It is given as:

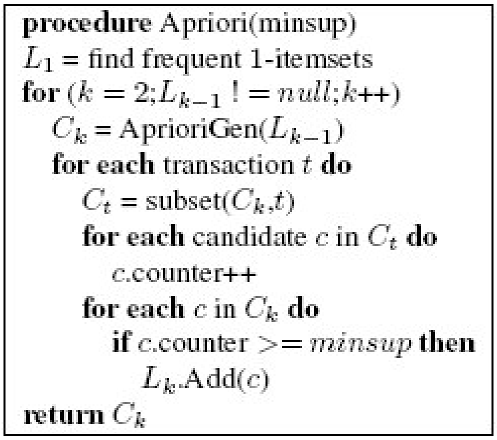
$$Completeness(WS^{\vee}, L_p) = \left(\frac{|\{s \mid s \in L_p \wedge s \models WS^{\vee}\}|}{|\{s \mid s \in L_p\}|} \right)$$

Equation H.2: Completeness

I: Pseudo code for Association Rule algorithms

1. Apriori algorithm

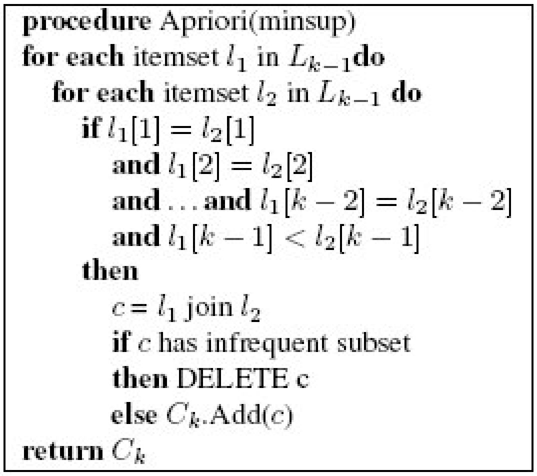
The Apriori algorithm [7] finds frequent itemsets using an iterative level-wise approach based on candidate generation. The input is a database of transactions and the minimum support count threshold. The pseudo code for the Apriori algorithm is given below:



```
procedure Apriori(minsup)
$L_1$ = find frequent 1-itemsets
for ($k = 2; L_{k-1} \;! = null; k$++)
   $C_k$ = AprioriGen($L_{k-1}$)
   for each transaction $t$ do
      $C_t$ = subset($C_k$,$t$)
      for each candidate $c$ in $C_t$ do
         $c$.counter++
      for each $c$ in $C_k$ do
         if $c$.counter $>= minsup$ then
            $L_k$.Add($c$)
return $C_k$
```

Figure I.1: Pseudo code of the Apriori algorithm

The Apriori algorithm calls the method AprioriGen (Figure I.2) to generate the candidate itemsets and then uses the Apriori property to eliminate those having an infrequent subset. The Apriori property states that: "Any (k-1) frequent itemset that is not frequent cannot be a subset of a frequent k-itemset. Hence, if any (k-1) subset of a candidate k-itemset is not in L_{k-1}, then the candidate cannot be frequent either and so can be removed from $C_{k.}$" The AprioriGen procedure performs two kinds of actions: join and prune. In the join component, L_{k-1} is joined with L_{k-1} to generate potential candidates. The prune component employs the Apriori property to remove candidates that have an infrequent subset.

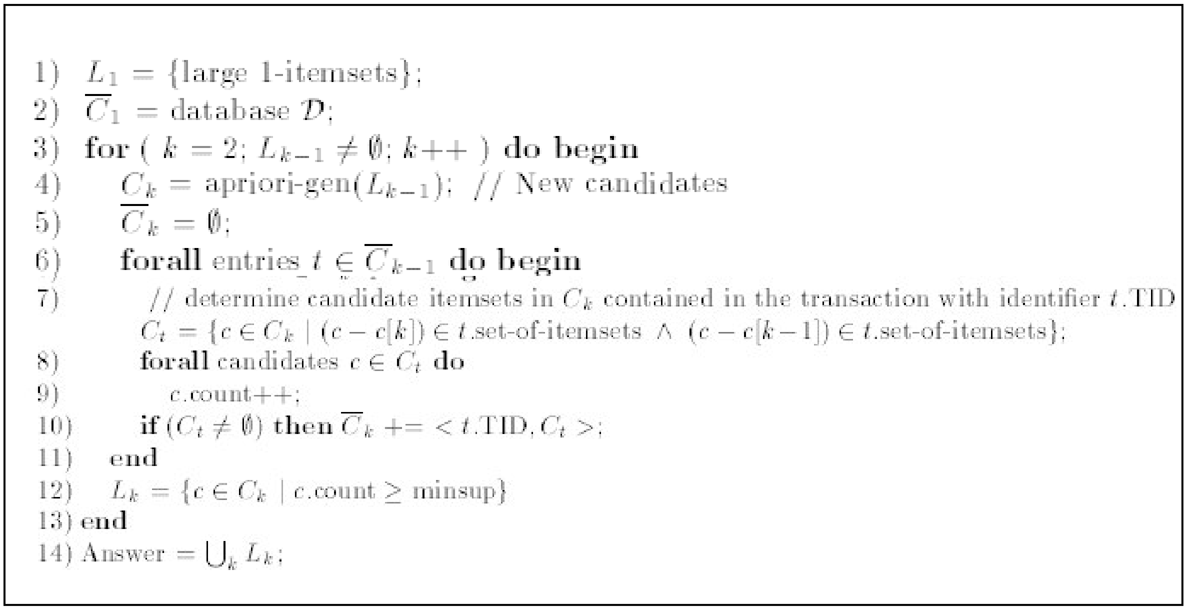


```
procedure Apriori(minsup)
for each itemset $l_1$ in $L_{k-1}$do
   for each itemset $l_2$ in $L_{k-1}$ do
      if $l_1[1] = l_2[1]$
         and $l_1[2] = l_2[2]$
         and ... and $l_1[k-2] = l_2[k-2]$
         and $l_1[k-1] < l_2[k-1]$
      then
         $c = l_1$ join $l_2$
         if $c$ has infrequent subset
         then DELETE c
         else $C_k$.Add($c$)
return $C_k$
```

Figure I.2: Pseudo code of the AprioriGen procedure used in the Apriori algorithm

2. AprioriTid algorithm

The AprioriTid algorithm [7] like the Apriori algorithm also uses the AprioriGen function (Figure I.2) to determine the candidate itemsets before the pass begins. But, the AprioriTid algorithm has an additional property that the database is not used at all for counting the support of candidate itemsets after the first pass. Rather than using the database transactions, this algorithm uses the entries in $\overline{c_k}$ to count the support of candidates in C_k. $\overline{c_k}$ is the set of candidate k-itemsets when the transaction IDs of the generating transactions are kept associated with the candidates. Keeping a track of transactions IDs from which the candidate frequent itemsets are generated at each level greatly reduces the reading effort in later passes. Once these candidate itemsets are obtained, association rules can be found just like in the Apriori algorithm. The pseudo code of the algorithm is given below in Figure I.3:

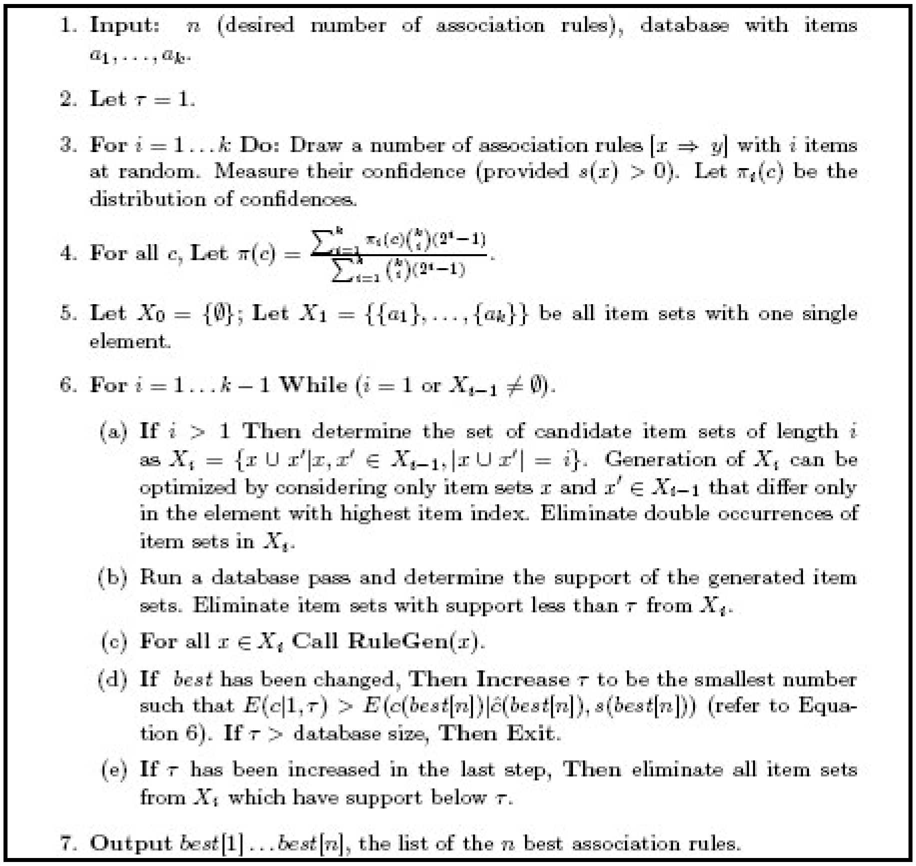


```
$L_1$ = {large 1-itemsets};
$\overline{C}_1$ = database $\mathcal{D}$;
for ( $k = 2$; $L_{k-1} \neq \emptyset$; $k$++ ) do begin
    $C_k$ = apriori-gen($L_{k-1}$); // New candidates
    $\overline{C}_k = \emptyset$;
    forall entries $t \in \overline{C}_{k-1}$ do begin
        // determine candidate itemsets in $C_k$ contained in the transaction with identifier $t$.TID
        $C_t = \{c \in C_k \mid (c - c[k]) \in t.\text{set-of-itemsets} \wedge (c - c[k-1]) \in t.\text{set-of-itemsets}\}$;
        forall candidates $c \in C_t$ do
            $c$.count++;
        if ($C_t \neq \emptyset$) then $\overline{C}_k$ += < $t$.TID, $C_t$ >;
    end
    $L_k = \{c \in C_k \mid c.\text{count} \geq \text{minsup}\}$
end
Answer = $\bigcup_k L_k$;
```

Figure I.3: AprioriTid algorithm

3. PredictiveApriori algorithm

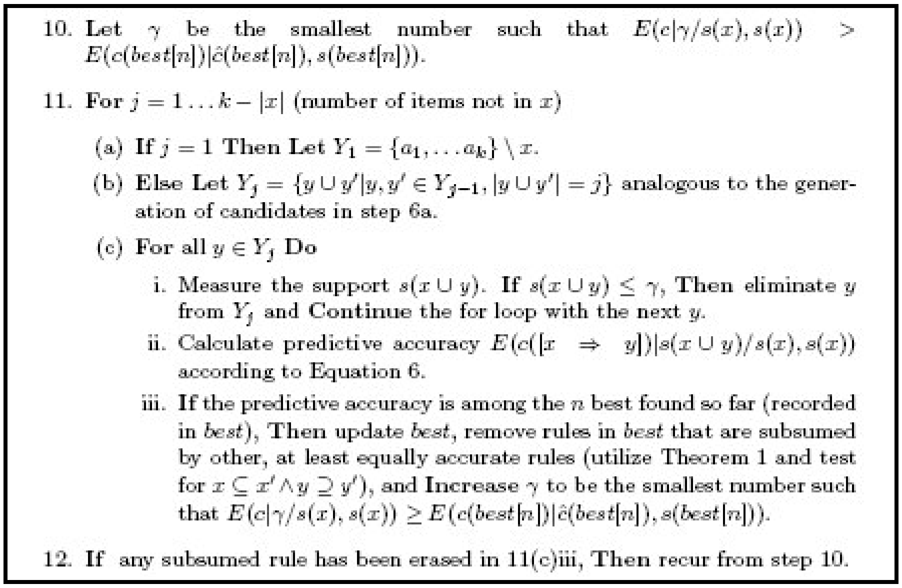
The Apriori algorithm finds association rules in two steps. First, all item sets *x* with support of more then the fixed threshold “minsup” are found. Then, all item sets are split into left and right hand side *x* and *y* (in all possible ways) and the confidence of the rules [*x* => *y*] is calculated as s (*x* U *y*)/ s(*x*). All rules with a confidence above the confidence threshold “minconf” are returned. The PredictiveApriori algorithm [22] differs from that scheme since we do not have fixed confidence and support thresholds. In fact, it discovers the best n rules. In the first step, the PredictiveApriori algorithm estimates the prior $\pi(c)$. Then generation of frequent item sets, pruning the hypothesis space by dynamically adjusting the minsup threshold, generating association rules, and removing redundant association rules interleave. The algorithm is displayed in Figure I.4 and the procedure for generation of all rules with fixed body *x* is presented in Figure I.5.



```
1. Input: $n$ (desired number of association rules), database with items $a_1, \ldots, a_k$.
2. Let $\tau = 1$.
3. For $i = 1 \ldots k$ Do: Draw a number of association rules $[x \Rightarrow y]$ with $i$ items at random. Measure their confidence (provided $s(x) > 0$). Let $\pi_i(c)$ be the distribution of confidences.
4. For all $c$, Let $\pi(c) = \frac{\sum_{i=1}^{k} \pi_i(c)\binom{k}{i}(2^i-1)}{\sum_{i=1}^{k}\binom{k}{i}(2^i-1)}$.
5. Let $X_0 = \{\emptyset\}$; Let $X_1 = \{\{a_1\}, \ldots, \{a_k\}\}$ be all item sets with one single element.
6. For $i = 1 \ldots k-1$ While ($i = 1$ or $X_{i-1} \neq \emptyset$).
    (a) If $i > 1$ Then determine the set of candidate item sets of length $i$ as $X_i = \{x \cup x' | x, x' \in X_{i-1}, |x \cup x'| = i\}$. Generation of $X_i$ can be optimized by considering only item sets $x$ and $x' \in X_{i-1}$ that differ only in the element with highest item index. Eliminate double occurrences of item sets in $X_i$.
    (b) Run a database pass and determine the support of the generated item sets. Eliminate item sets with support less than $\tau$ from $X_i$.
    (c) For all $x \in X_i$ Call RuleGen($x$).
    (d) If $best$ has been changed, Then Increase $\tau$ to be the smallest number such that $E(c|1,\tau) > E(c(best[n])|\hat{c}(best[n]), s(best[n]))$ (refer to Equation 6). If $\tau >$ database size, Then Exit.
    (e) If $\tau$ has been increased in the last step, Then eliminate all item sets from $X_i$ which have support below $\tau$.
7. Output $best[1] \ldots best[n]$, the list of the $n$ best association rules.
```

Figure I.4: PredictiveApriori algorithm

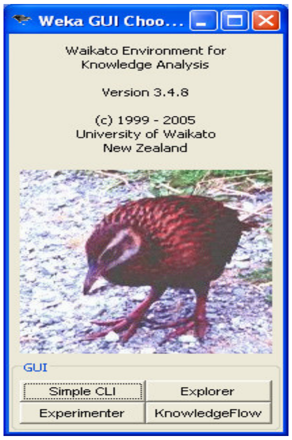
The procedure RuleGen(*x*) finds the best rules with body *x* and is given below:



```
10. Let $\gamma$ be the smallest number such that $E(c|\gamma/s(x), s(x)) > E(c(best[n])|\hat{c}(best[n]), s(best[n]))$.
11. For $j = 1 \ldots k - |x|$ (number of items not in $x$)
    (a) If $j = 1$ Then Let $Y_1 = \{a_1, \ldots a_k\} \setminus x$.
    (b) Else Let $Y_j = \{y \cup y' | y, y' \in Y_{j-1}, |y \cup y'| = j\}$ analogous to the generation of candidates in step 6a.
    (c) For all $y \in Y_j$ Do
        i. Measure the support $s(x \cup y)$. If $s(x \cup y) \leq \gamma$, Then eliminate $y$ from $Y_j$ and Continue the for loop with the next $y$.
        ii. Calculate predictive accuracy $E(c([x \Rightarrow y])|s(x \cup y)/s(x), s(x))$ according to Equation 6.
        iii. If the predictive accuracy is among the $n$ best found so far (recorded in $best$), Then update $best$, remove rules in $best$ that are subsumed by other, at least equally accurate rules (utilize Theorem 1 and test for $x \subseteq x' \wedge y \supseteq y'$), and Increase $\gamma$ to be the smallest number such that $E(c|\gamma/s(x), s(x)) \geq E(c(best[n])|\hat{c}(best[n]), s(best[n]))$.
12. If any subsumed rule has been erased in 11(c)iii, Then recur from step 10.
```

Figure I.5: The procedure RuleGen

J: The Weka library

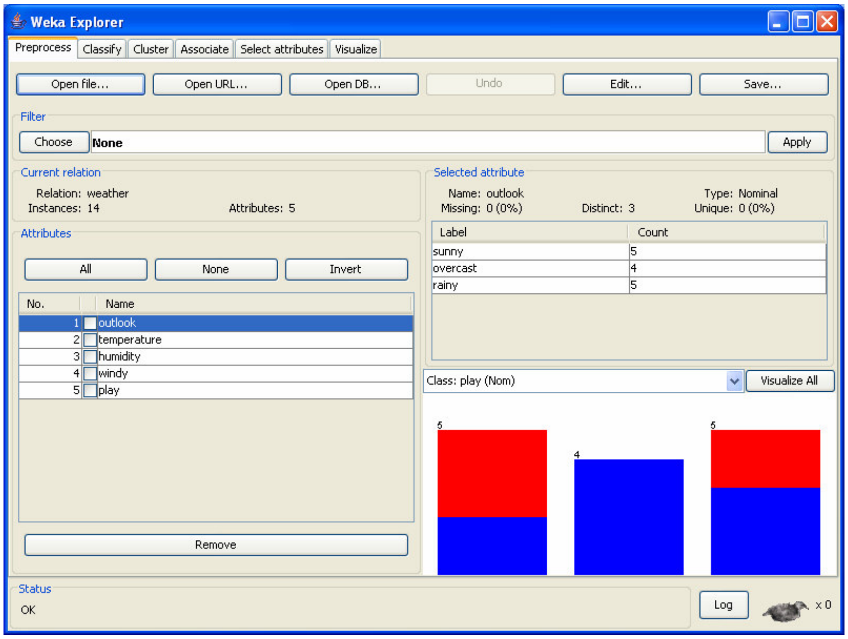


Waikato Environment for
Knowledge Analysis

Version 3.4.8

(c) 1999 - 2005
University of Waikato
New Zealand

Data mining includes tasks such as classification, estimation, prediction, affinity grouping, clustering, and description and profiling tasks. The Weka workbench provides tools for performing all these tasks. In this appendix we give a brief overview of the Weka library. Following (Figure J.1) screenshot from Weka shows all the mining options available in it. We can see that we have loaded a file containing weather information like temperature, humidity, wind etc. For any algorithm we can also select the number of attributes, we have a choice to deselect some attributes or use all of them. In the figure it can be seen that it is possible to have the information like number of records in the input file (like in this example, it is 14). At the right hand side we see that the weather outlook can have values like hot, mild and cool and in how many records these values occur. In the menu bar we can see mining options for classification, clustering, and association rule mining. Readers are referred to [28] for detailed reading about each of these mining options.



Filter: None

Relation: weather
Instances: 14
Attributes: 5

Name: outlook
Type: Nominal
Missing: 0 (0%)
Distinct: 3
Unique: 0 (0%)

Label	Count
sunny	5
overcast	4
rainy	5

No.		Name
1	□	outlook
2	□	temperature
3	□	humidity
4	□	windy
5	□	play

Class: play (Nom)

Status: OK

Figure J.1: The Weka Workbench's Explorer

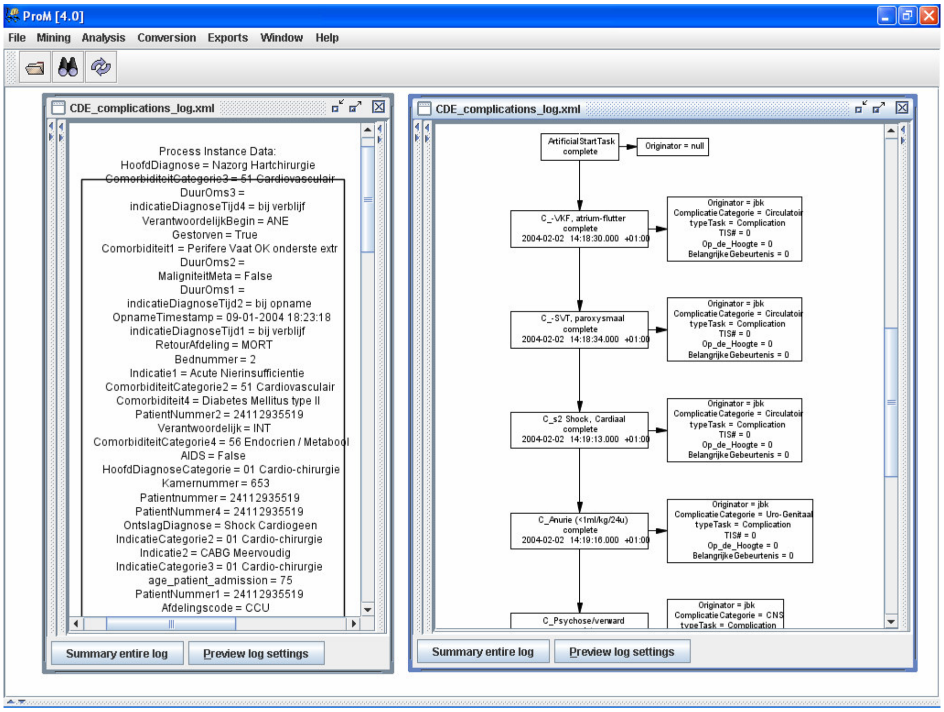
K: Case Data Extraction mining plug-in

1. Case properties

The CDE plug-in as the name indicates extracts case properties of the event log loaded in the framework. These properties may be:

- Process instance data attributes
- Audit trail entry data attributes
- Originators
- Event type

To understand these various cases properties let us take an example healthcare log. The log has 6 PIs and 18 ATEs. The PIs of the log refer to complications that patients can suffer from. Using the *Preview log settings* option in ProM we take a look at the different data attributes for this log. Following figure shows the properties of the process instance as well as the properties of the activities/tasks in the process instance. The process instance data attributes are shown at the left hand side and the data elements for the audit trail entries are shown at the right hand side.



```
Process Instance Data:
HoofdDiagnose = Nazorg Hartchirurgie
ComorbiditeitCategorie3 = 51 Cardiovasculair
DuurOms3 =
indicatieDiagnoseTijd4 = bij verblijf
VerantwoordelijkBegin = ANE
Gestorven = True
Comorbiditeit1 = Perifere Vaat OK onderste extr
DuurOms2 =
MaligniteitMeta = False
DuurOms1 =
indicatieDiagnoseTijd2 = bij opname
OpnameTimestamp = 09-01-2004 18:23:18
indicatieDiagnoseTijd1 = bij verblijf
RetourAfdeling = MORT
Bednummer = 2
Indicatie1 = Acute Nierinsufficientie
ComorbiditeitCategorie2 = 51 Cardiovasculair
Comorbiditeit4 = Diabetes Mellitus type II
PatientNummer2 = 24112935519
Verantwoordelijk = INT
ComorbiditeitCategorie4 = 56 Endocrien / Metabool
AIDS = False
HoofdDiagnoseCategorie = 01 Cardio-chirurgie
Kamernummer = 653
Patientnummer = 24112935519
PatientNummer4 = 24112935519
OntslagDiagnose = Shock Cardiogeen
IndicatieCategorie2 = 01 Cardio-chirurgie
Indicatie2 = CABG Meervoudig
IndicatieCategorie3 = 01 Cardio-chirurgie
age_patient_admission = 75
PatientNummer1 = 24112935519
Afdelingscode = CCU
```

```
ArtificialStartTask
complete
Originator = null

C_-VKF, atrium-flutter
complete
2004-02-02  14:18:30.000  +01:00
Originator = jbk
ComplicatieCategorie = Circulatoir
typeTask = Complication
TIS# = 0
Op_de_Hoogte = 0
BelangrijkeGebeurtenis = 0

C_-SVT, paroxysmaal
complete
2004-02-02  14:18:34.000  +01:00
Originator = jbk
ComplicatieCategorie = Circulatoir
typeTask = Complication
TIS# = 0
Op_de_Hoogte = 0
BelangrijkeGebeurtenis = 0

C_s2 Shock, Cardiaal
complete
2004-02-02  14:19:13.000  +01:00
Originator = jbk
ComplicatieCategorie = Circulatoir
typeTask = Complication
TIS# = 0
Op_de_Hoogte = 0
BelangrijkeGebeurtenis = 0

C_Anurie (<1ml/kg/24u)
complete
2004-02-02  14:19:16.000  +01:00
Originator = jbk
ComplicatieCategorie = Uro-Genitaal
typeTask = Complication
TIS# = 0
Op_de_Hoogte = 0
BelangrijkeGebeurtenis = 0

C_Psychose/verward
Originator = jbk
ComplicatieCategorie = CNS
typeTask = Complication
```

Figure K.1: Process instance and ATE data elements

In this figure, the process instance data elements can be seen, these includes properties like 'HoofdDiagnose', 'Gestorven', 'Bednummer', 'IndicatieCategorie6' etc. These all are properties of the process instance which is a patient in this case. For example, the information about his main diagnosis is captured in the attribute 'HoofdDiagnose', his bed number information is stored in the attribute 'Bednummer' etc. The data attributes for audit trail entries i.e. the complications this particular patient suffers from includes the properties like 'Originator', 'ComplicatieCategory', 'TIS#' etc. These data attributes store information about the complications. For example, the complication category information is stored in the attribute 'ComplicatieCategory'. It is interesting to look at these case properties as they help to gain insight into the process through the various data attributes related to the case itself or to the various tasks in the process. In the next section the CDE plug-in is explained.

2. The CDE plug-in

The CDE mining plug-in allows a user to convert the case data (case properties) into a table which can be imported to a spreadsheet like MS-Excel in the form of CSV file. CSV is sometimes also called Comma Delimited format.

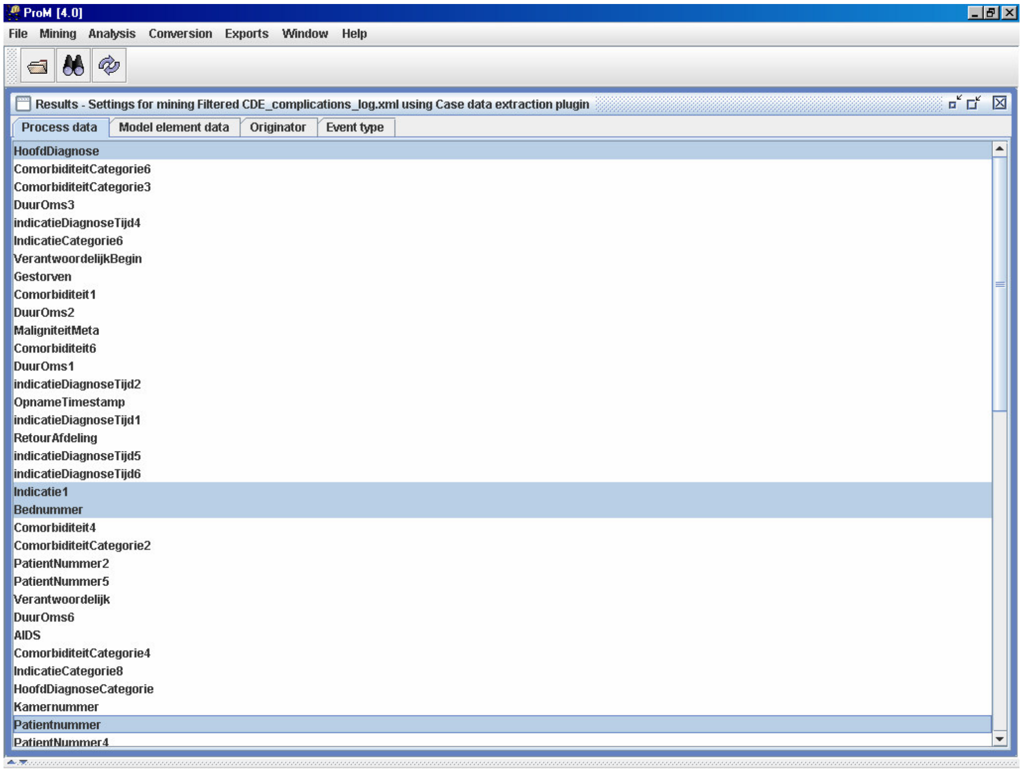
Before converting the data from a log into a table, the user can make his selections from the data attributes of the process instances in the log, the data attributes of various activities in the log, originators and event types. Once the user has made his selection, for each process instance the table includes [38]:

- its name,
- an estimation of its makespan, and
- an overview of its selected data fields.

For every ATE, the table includes:

- the number of times a selected event occurred,
- its service time,
- an overview of its selected originators, and
- an overview of its selected data fields.

This table can be then exported to a CSV file using the standard CSV Export plug-in available in the ProM. These CSV files can be then imported in Microsoft Excel. We now understand how the CDE and CSV plug-ins function with the help of an example. The log displayed in Figure K.1 is used again. The result of mining this log using the CDE plug-in is as shown in the Figure K.2.

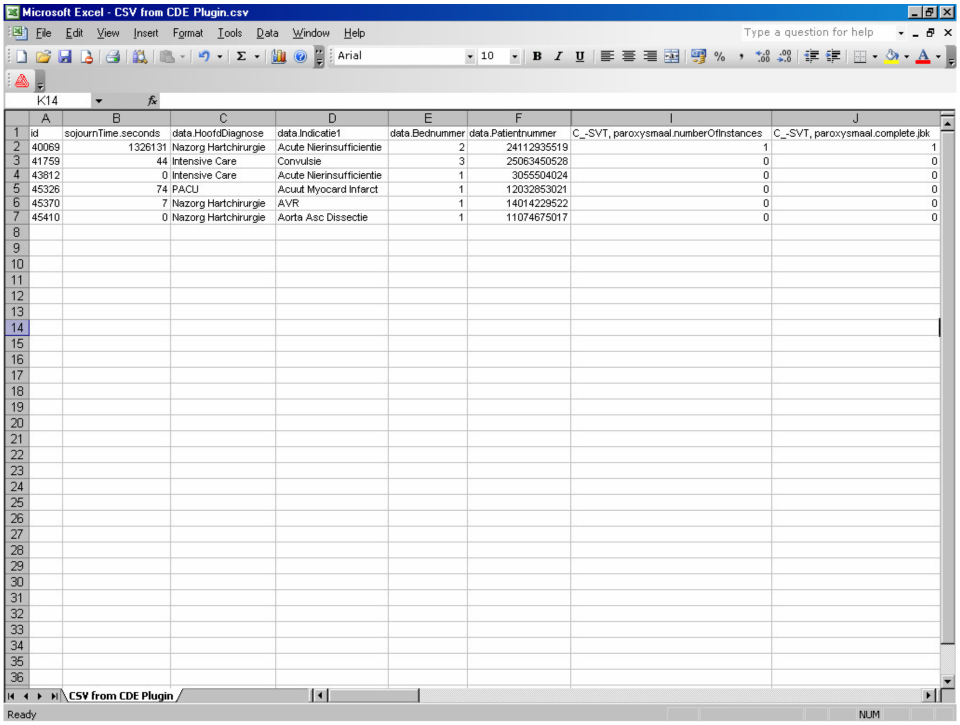


Results - Settings for mining Filtered CDE_complications_log.xml using Case data extraction plugin

HoofdDiagnose
ComorbiditeitCategorie6
ComorbiditeitCategorie3
DuurOms3
indicatieDiagnoseTijd4
IndicatieCategorie6
VerantwoordelijkBegin
Gestorven
Comorbiditeit1
DuurOms2
MaligniteitMeta
Comorbiditeit6
DuurOms1
indicatieDiagnoseTijd2
OpnameTimestamp
indicatieDiagnoseTijd1
RetourAfdeling
indicatieDiagnoseTijd5
indicatieDiagnoseTijd6
Indicatie1
Bednummer
Comorbiditeit4
ComorbiditeitCategorie2
PatientNummer2
PatientNummer5
Verantwoordelijk
DuurOms6
AIDS
ComorbiditeitCategorie4
IndicatieCategorie8
HoofdDiagnoseCategorie
Kamernummer
Patientnummer
PatientNummer4

Figure K.2: CDE plug-in outputs a table displaying all data attributes for the case and ATEs

As seen, the output is a table showing the case properties in the column *Process* Data, the audit trail entries properties in the column *Model Element Data*, the list of originators in the column *Originators*, and the event type information in the column *Event type*. The user can select different attributes and his selection is indicated by a grey shade. As we can see in the Figure K.2, the process data properties- HoofdDiagnose (main diagnosis of the patient), Indicatie1 (patient's main indication i.e. his complication), Bednummer (bed number assigned to the patient), HoofdDiagnosecategories (the category of the patient's main diagnosis) and PatientNummer (number assigned to the patient) is selected. These selected data attributes can be now exported to a CSV format using the CSV export plug-in. The result of using this plug-in is shown in the Figure K.3.


Microsoft Excel - CSV from CDE Plugin.csv
File Edit View Insert Format Tools Data Window Help
Type a question for help
Arial
10
K14
CSV from CDE Plugin
Ready
NUM

	A	B	C	D	E	F	I	J
1	id	sojournTime.seconds	data.HoofdDiagnose	data.Indicatie1	data.Bednummer	data.Patientnummer	C_-SVT, paroxysmaal.numberOfInstances	C_-SVT, paroxysmaal.complete.jbk
2	40069	1326131	Nazorg Hartchirurgie	Acute Nierinsufficientie	2	24112935519	1	1
3	41759	44	Intensive Care	Convulsie	3	25063450528	0	0
4	43812	0	Intensive Care	Acute Nierinsufficientie	1	3055504024	0	0
5	45326	74	PACU	Acuut Myocard Infarct	1	12032853021	0	0
6	45370	7	Nazorg Hartchirurgie	AVR	1	14014229522	0	0
7	45410	0	Nazorg Hartchirurgie	Aorta Asc Dissectie	1	11074675017	0	0

Figure K.3: CSV file from CDE plug-in exported to MS-Excel

Figure K.3 provides us the following information:

- Process Instance identifier: The first column (column A) is the process instance identifier for the process instances in the event log. This shows the case being handled. Every row corresponds to one unique process identifier. In turn, one process instance can correspond to multiple audit trail entries, for example we can see the ATE *C_-SVT Paroxysmaal* in column I. In figure 3.30 we have process instance identifier as 40069, 41759 etc.
- Sojourn time: Sojourn Time is the maximal logged event time minus the minimal logged event time for this instance. This can be seen in column B as sojournTime.seconds.
- data.D: Corresponding to each process instance an overview of its selected data attributes is given. We can see the value of the process instance-level data element D. In Figure 4.2 for example, we chose the process instance data attribute 'hoofdDiagnose', and we can see the values for this attribute in column C. Values for other case properties are present in columns D, E and F.
- T.numberOfInstances: The number of instances in which a task (ATE) appears is also included in the CSV file. For example in the above figure, the task C_SVT appears once in the process instance 40069 whereas it is not at all executed in the process instance 41759 and others as seen by a value 0 in the column I.
- T.lowServiceTime.seconds: For task T its minimal service time in seconds is also displayed in the CSV.
- T.highServiceTime.seconds: Like the minimal service time for a task T, the CSV file also gives the maximal service time in seconds for the task T.
- T.E.U: T.E.U gives the number of times user U has caused event E for task T.
- T.data.D: This gives the value of the (first) task-instance level data element D. For example, we see the originator data in column J which indicates that the originator 'jbk' has executed process instance 40069 and not any other activity. Other selected data attributes are also present in further columns.

This CSV file exported from the CDE plug-in gives us insights into the process underlying this event log. For example, we can take a quick look at all the data attributes of the case and the activities in the process. We can have the information about which originator performed which activity and for which case. The information about which case has a particular activity can also be obtained. All these information can be put to use for different goals. For example, we can have a preliminary idea about the idleness of an originator as we can see for which case and activity he is engaged, and where he is idle. Below we discuss some application areas where the output of the case data extraction plug-in is useful.

1. As also mentioned earlier, general overview about process based on its data is obtained. This gives us insight into the process execution.
2. We can make some early predictions about the performance of the process based on the timestamp data attribute. We can be benefited by sojourn time and service time values. Based on these values we can accordingly alter the flow of process or change the allocation of resources.
3. We can also make early estimations about resource utilization. For example, if we see an originator 'jbk' always performs the activity 'A' we can assume he is a specialist in context of this activity and we can allocate him the tasks based on this understanding. We can also see if a resource is doing many activities or none of the activities in that process, this gives us a raw indication of his idle time.
4. We can also provide stakeholders with some valuable information about the tasks based on the values of their data elements. For example, in Figure K.3, we see three process instances data element 'data.HoofdDiagnose' has value 'Nazorg Hartchirurgie' out of 6 cases being handled in this process. This indicated that more patients are diagnosed with this complication and hence the personnel at the hospital must be well prepared to handle this complication. The benefits mentioned in 2, 3, and 4 serve as raw and early indications to begin the analysis with. They can be further confirmed with some other tools and techniques.
5. Besides, the benefits mentioned above, the output from the CDE plug-in can be used as input to other mining tools. In fact, the CDE plug-in was implemented in order to provide input to tools like Viscovery and NetMiner. The output can also be used in experimenting with machine learning algorithms provided in the data mining tool WEKA.

L: Using CSV from ProM for Weka

As already mentioned that the output of CDE can be exported to CSV format, and this CSV file can be used for our experiments for data mining tasks in the Weka library. This is so because Weka automatically converts a CSV file into its native data storage method: the ARFF format. The ARFF consists of a list of instances and attribute values for each instance separated by commas. CSV can be easily converted to ARFF format. This appendix explains how a CSV file can be converted to ARFF. For this consider the following CSV file generated from the CDE plug-in in combination with the CSV export plug-in:

```
PID,A.numInstances,B.numInstances,C.numInstances,D.numInstances

40067,2,0,1,1
40068,0,0,1,0
40069,1,0,1,0
40070,1,1,1,0
```

Figure L.1: An example CSV file in a text editor

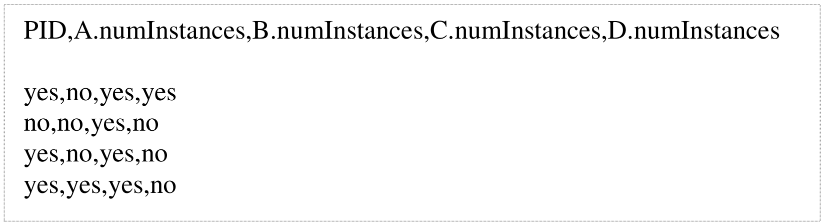
In the above figure, we see five data attributes, viz., PID and number of instances in which tasks A, B, C and D occurs. The next four rows show the four PIs with their values for these data attributes. To convert it into ARFF format, we add the name of the dataset using the @relation tag, the attribute information using the @attribute, and a @data line. This is shown in the following figure:

```
@relation CSV_to_ARFF

@attribute PID {….}
@attribute A_numberOfInstances {….}
@attribute B_numberOfInstances {….}
@attribute C_numberOfInstances {….}
@attribute D_numberOfInstances {….}

@data
40062, 2,0,1,1
40068, 0,0,1,0
40069, 1,1,1,0
```

Figure L.2: ARFF file

We can see the name of the dataset in the @relation tag, and the three data attributes are declared using the @attribute tags. The curly braces {…} are for the values that these attributes can have. Finally the process instances along with values for the data attributes are shown under @data tag. We show this conversion here for the purpose of understanding, however for experimenting with Weka, we need not manually convert a CSV file to ARFF format. *This conversion is automatically done by Weka*. So, we can simply give a CSV file as an input to the Weka workbench. But the CSV file in Figure L.1 can't be directly used in Weka for association analysis algorithms as these algorithms need binary information for generating Boolean association rules. So we first need to change the CSV file and then load it again into Weka

In Figure L.3, we have all numeric data. To be able to use this for association rule mining this numeric data has to be converted to strings (the association analysis algorithms can not handle numeric values). For this, each presence of a task in a PI is replaced by a yes and its absence by no. It means the value that the attributes can hold is only a yes or no. The column PID can be removed as each row uniquely defines a new PID. So we have:



```
PID,A.numInstances,B.numInstances,C.numInstances,D.numInstances

yes,no,yes,yes
no,no,yes,no
yes,no,yes,no
yes,yes,yes,no
```

Figure L.4: We replace the numeric data values by yes/no

Now this file can be used for association analysis algorithms available in the Weka library. This is how the CSV file obtained from ProM can be modified to be used for association rule algorithms in Weka.

M: Case study2

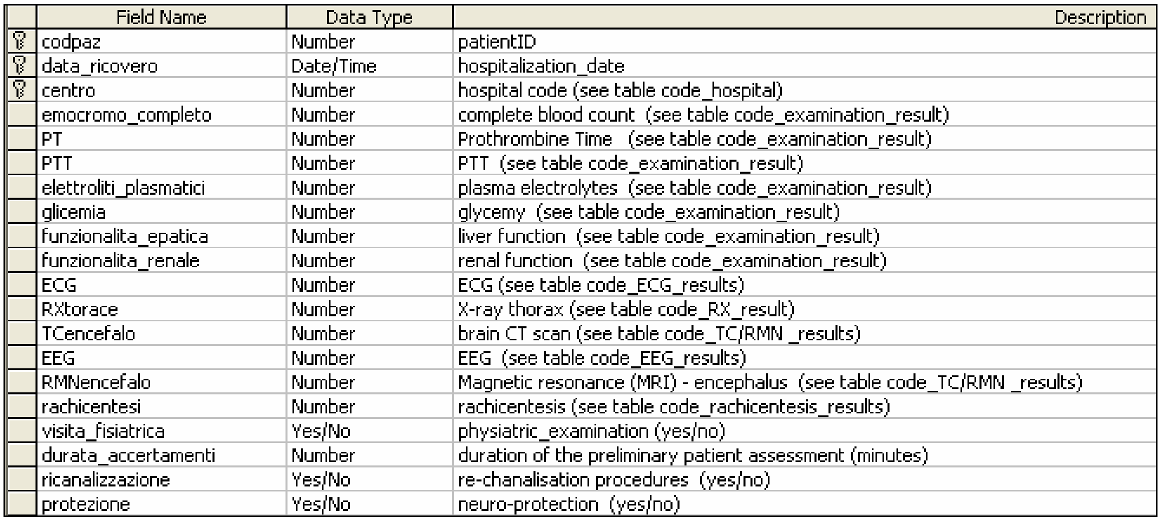
Data in Case study2 is organized in form of tables in MS-Access database. In this appendix, we provide all the important tables in this database. This gives us insights into what data is stored in the database and in what format. The data in these tables pertains to patient's general information, his stroke symptoms during admission, previous history, therapies given to him, measurements he undergoes etc. However, we do not show this data because most of it is in form of numbers (codes taken up from some other tables). The tables in the database are organized as given in Chapter 7, Figure 7.1:

- A patient’s personal data is recorded in the table *data_personal data* and this table is linked to every other table in the database as this table gives identification of any patient.
- For every patient:
    - Data about his clinical history is recorded in the table *data_anamnesis.*
    - data about previous therapies given to him is recorded in the table *data_anamnesis_therapies*,
    - Data about his measurements for various life parameters like pulse rate, body temperature etc. is recorded in table *data_life_parameters*.
    - Data pertaining to his hospitalization i.e. results of various examinations a patient undergoes while examination and data pertaining to his discharge is recorded in the tables: *data_neurological examination, data_objective_examination* and *data_admission_discharge*. During the hospirtalization, details are also stored about what happened before the patient came to the hospital (before he is admitted). These details are stored in the table *data_pre_hospital_phase,*
    - Data related to monitoring and his treatment at the hospital pertains to the various therapies he is given and this information is recorded in tables for different therapies (*data_physical_therapy*, *data_therapy_acute_phase, data_therapy_subacute_phase*), and the complications the patients suffers from is recorded in the table *data_medical_complications.*
    - If the patient is admitted within 6 hours from the stroke patients, his data is recorded in the table *data_acute_phase* and the table *data_subacute_phase* stores data about a patient admitted after the first 6 hours of stroke symptoms.
    - Stroke patients undergo one or more kinds of measurements. These are done on scales identified as: Barthel, Glasgow_coma, Hamilton_anxiety, Hamilton_depression, London_handicap, SF36 and NIH. The data about the patient’s measurements on these scales are recorded in the respective tables for these measurements.
    - Once the patient is discharged the follow-up period starts and during this period his data is recorded in the table *data_followup, data_visits_during_follow_up* and in case the patient is hospitalized in this period his date is stored in the table *data_re-hospitalization_during_follow_up*.

For each of these tables mentioned above, we give the columns of the table (the structure of the table in form of its fields) and the number of records in the table. Structure of any table includes *field name, data type and description.* Field name is the name of the column in the data table, data type tells us what kind of data this field can hold i.e. number or text or Boolean etc. Description about the field is a short remark about the field. It is optional. In Appendix E, the readers can know which of these tables were used for making event logs for experiments done in this graduation project.

Data_acute_phase

This table stores details about any patient’s state during the acute phase. This table uses pre-defined codes for most of the parameters that define the state of any patient during acute phase, therefore this table refers to many other tables (identified by *code_* , but we do not show these tables). The total number of records in this table is 114.

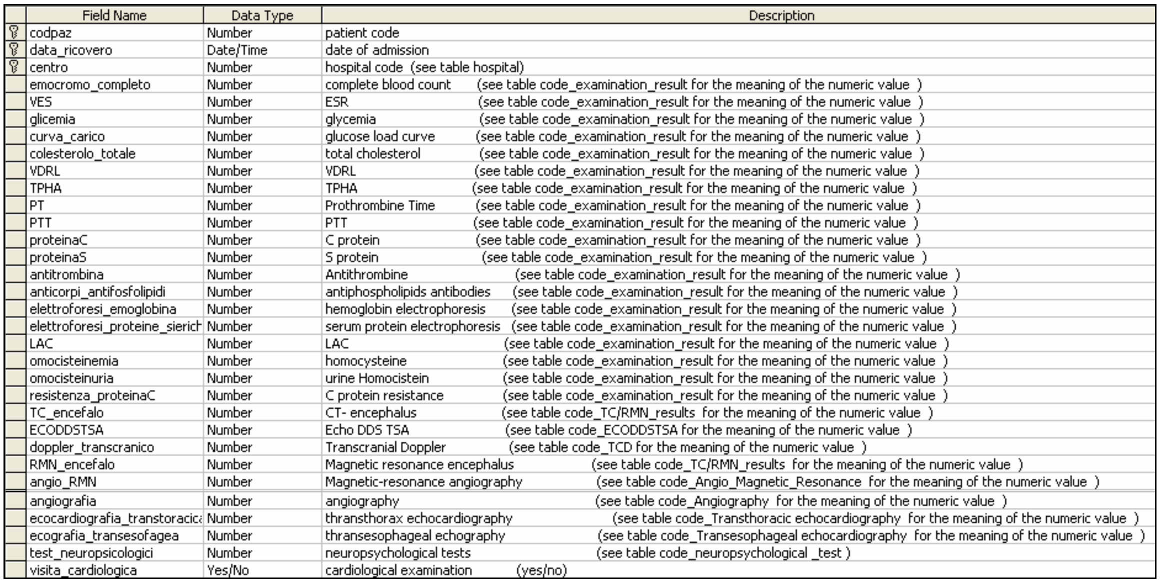


	Field Name	Data Type	Description
🔑	codpaz	Number	patientID
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
	emocromo_completo	Number	complete blood count (see table code_examination_result)
	PT	Number	Prothrombine Time (see table code_examination_result)
	PTT	Number	PTT (see table code_examination_result)
	elettroliti_plasmatici	Number	plasma electrolytes (see table code_examination_result)
	glicemia	Number	glycemy (see table code_examination_result)
	funzionalita_epatica	Number	liver function (see table code_examination_result)
	funzionalita_renale	Number	renal function (see table code_examination_result)
	ECG	Number	ECG (see table code_ECG_results)
	RXtorace	Number	X-ray thorax (see table code_RX_result)
	TCencefalo	Number	brain CT scan (see table code_TC/RMN _results)
	EEG	Number	EEG (see table code_EEG_results)
	RMNencefalo	Number	Magnetic resonance (MRI) - encephalus (see table code_TC/RMN _results)
	rachicentesi	Number	rachicentesis (see table code_rachicentesis_results)
	visita_fisiatrica	Yes/No	physiatric_examination (yes/no)
	durata_accertamenti	Number	duration of the preliminary patient assessment (minutes)
	ricanalizzazione	Yes/No	re-chanalisation procedures (yes/no)
	protezione	Yes/No	neuro-protection (yes/no)

Figure M.1: Structure of Data_acute_phase table

Data_subacute_phase

This table stores details about any patient's state during the sub-acute phase. This table uses pre-defined codes for most of the parameters that define the state of any patient during sub-acute phase, therefore this table refers to many other tables (identified by *code_* , but we do not show these tables). The total number of records in this table is 378.

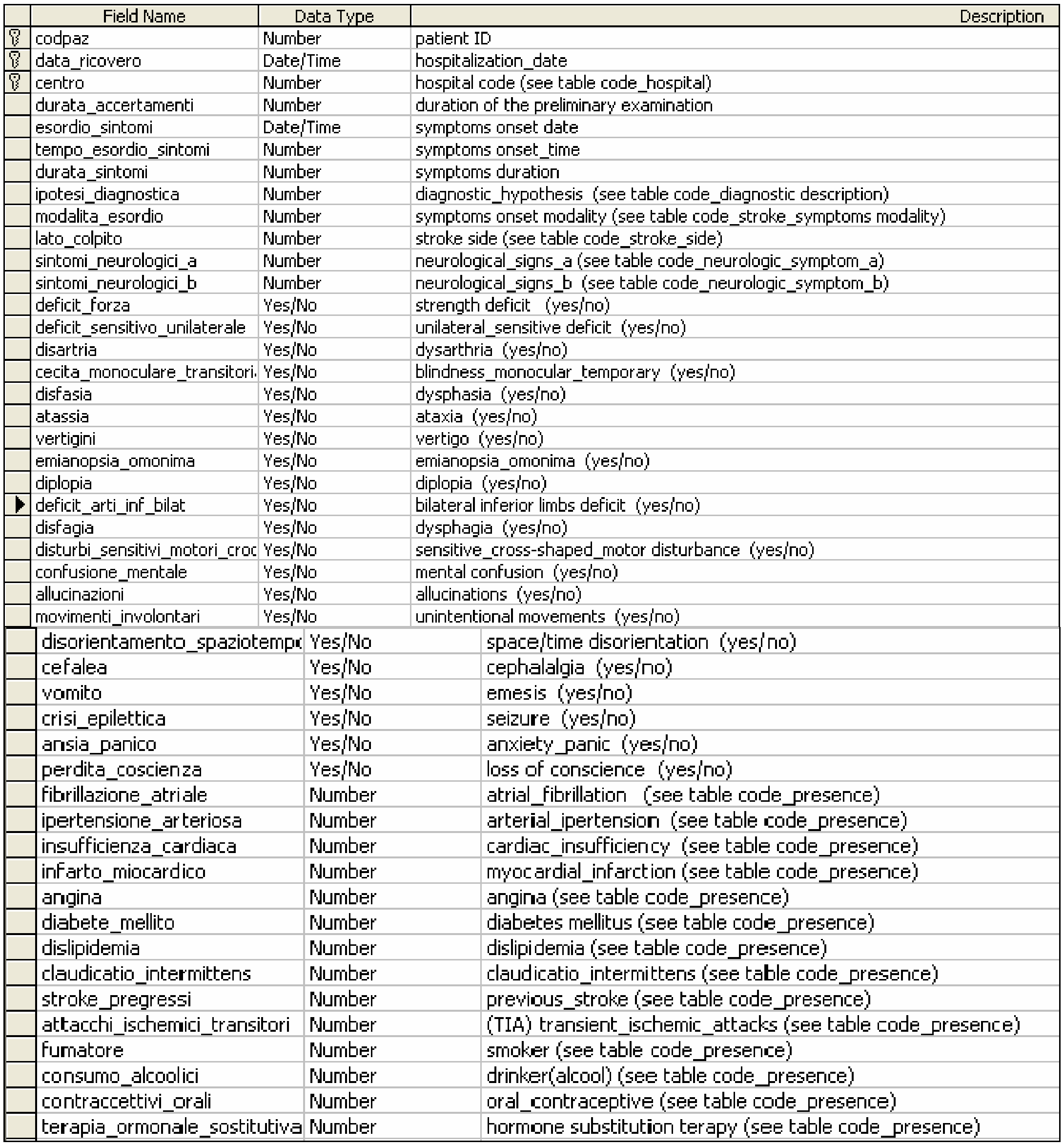


	Field Name	Data Type	Description
🔑	codpaz	Number	patient code
🔑	data_ricovero	Date/Time	date of admission
🔑	centro	Number	hospital code (see table hospital)
	emocromo_completo	Number	complete blood count (see table code_examination_result for the meaning of the numeric value)
	VES	Number	ESR (see table code_examination_result for the meaning of the numeric value)
	glicemia	Number	glycemia (see table code_examination_result for the meaning of the numeric value)
	curva_carico	Number	glucose load curve (see table code_examination_result for the meaning of the numeric value)
	colesterolo_totale	Number	total cholesterol (see table code_examination_result for the meaning of the numeric value)
	VDRL	Number	VDRL (see table code_examination_result for the meaning of the numeric value)
	TPHA	Number	TPHA (see table code_examination_result for the meaning of the numeric value)
	PT	Number	Prothrombine Time (see table code_examination_result for the meaning of the numeric value)
	PTT	Number	PTT (see table code_examination_result for the meaning of the numeric value)
	proteinaC	Number	C protein (see table code_examination_result for the meaning of the numeric value)
	proteinaS	Number	S protein (see table code_examination_result for the meaning of the numeric value)
	antitrombina	Number	Antithrombine (see table code_examination_result for the meaning of the numeric value)
	anticorpi_antifosfolipidi	Number	antiphospholipids antibodies (see table code_examination_result for the meaning of the numeric value)
	elettroforesi_emoglobina	Number	hemoglobin electrophoresis (see table code_examination_result for the meaning of the numeric value)
	elettroforesi_proteine_sierich	Number	serum protein electrophoresis (see table code_examination_result for the meaning of the numeric value)
	LAC	Number	LAC (see table code_examination_result for the meaning of the numeric value)
	omocisteinemia	Number	homocysteine (see table code_examination_result for the meaning of the numeric value)
	omocisteinuria	Number	urine Homocistein (see table code_examination_result for the meaning of the numeric value)
	resistenza_proteinaC	Number	C protein resistance (see table code_examination_result for the meaning of the numeric value)
	TC_encefalo	Number	CT- encephalus (see table code_TC/RMN_results for the meaning of the numeric value)
	ECODDSTSA	Number	Echo DDS TSA (see table code_ECODDSTSA for the meaning of the numeric value)
	doppler_transcranico	Number	Transcranial Doppler (see table code_TCD for the meaning of the numeric value)
	RMN_encefalo	Number	Magnetic resonance encephalus (see table code_TC/RMN_results for the meaning of the numeric value)
	angio_RMN	Number	Magnetic-resonance angiography (see table code_Angio_Magnetic_Resonance for the meaning of the numeric value)
	angiografia	Number	angiography (see table code_Angiography for the meaning of the numeric value)
	ecocardiografia_transtoracica	Number	thransthorax echocardiography (see table code_Transthoracic echocardiography for the meaning of the numeric value)
	ecografia_transesofagea	Number	thransesophageal echography (see table code_Transesophageal echocardiography for the meaning of the numeric value)
	test_neuropsicologici	Number	neuropsychological tests (see table code_neuropsychological _test)
	visita_cardiologica	Yes/No	cardiological examination (yes/no)

Figure M.2: Structure of Data_subacute_phase table

Data_anamnesis

This table stores details about the previous medical history of the patient. The total number of records in this table is 385.

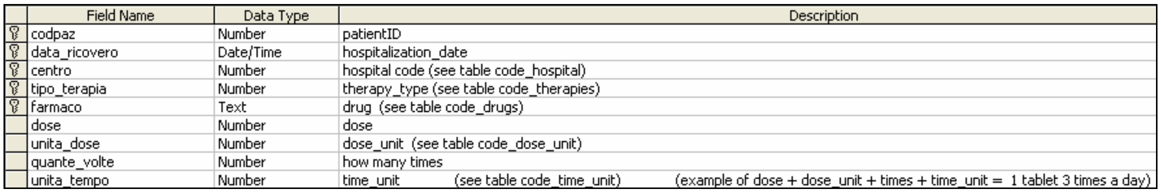


	Field Name	Data Type	Description
🔑	codpaz	Number	patient ID
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
	durata_accertamenti	Number	duration of the preliminary examination
	esordio_sintomi	Date/Time	symptoms onset date
	tempo_esordio_sintomi	Number	symptoms onset_time
	durata_sintomi	Number	symptoms duration
	ipotesi_diagnostica	Number	diagnostic_hypothesis (see table code_diagnostic description)
	modalita_esordio	Number	symptoms onset modality (see table code_stroke_symptoms modality)
	lato_colpito	Number	stroke side (see table code_stroke_side)
	sintomi_neurologici_a	Number	neurological_signs_a (see table code_neurologic_symptom_a)
	sintomi_neurologici_b	Number	neurological_signs_b (see table code_neurologic_symptom_b)
	deficit_forza	Yes/No	strength deficit (yes/no)
	deficit_sensitivo_unilaterale	Yes/No	unilateral_sensitive deficit (yes/no)
	disartria	Yes/No	dysarthria (yes/no)
	cecita_monoculare_transitori	Yes/No	blindness_monocular_temporary (yes/no)
	disfasia	Yes/No	dysphasia (yes/no)
	atassia	Yes/No	ataxia (yes/no)
	vertigini	Yes/No	vertigo (yes/no)
	emianopsia_omonima	Yes/No	emianopsia_omonima (yes/no)
	diplopia	Yes/No	diplopia (yes/no)
▶	deficit_arti_inf_bilat	Yes/No	bilateral inferior limbs deficit (yes/no)
	disfagia	Yes/No	dysphagia (yes/no)
	disturbi_sensitivi_motori_croc	Yes/No	sensitive_cross-shaped_motor disturbance (yes/no)
	confusione_mentale	Yes/No	mental confusion (yes/no)
	allucinazioni	Yes/No	allucinations (yes/no)
	movimenti_involontari	Yes/No	unintentional movements (yes/no)
	disorientamento_spaziotemp	Yes/No	space/time disorientation (yes/no)
	cefalea	Yes/No	cephalalgia (yes/no)
	vomito	Yes/No	emesis (yes/no)
	crisi_epilettica	Yes/No	seizure (yes/no)
	ansia_panico	Yes/No	anxiety_panic (yes/no)
	perdita_coscienza	Yes/No	loss of conscience (yes/no)
	fibrillazione_atriale	Number	atrial_fibrillation (see table code_presence)
	ipertensione_arteriosa	Number	arterial_ipertension (see table code_presence)
	insufficienza_cardiaca	Number	cardiac_insufficiency (see table code_presence)
	infarto_miocardico	Number	myocardial_infarction (see table code_presence)
	angina	Number	angina (see table code_presence)
	diabete_mellito	Number	diabetes mellitus (see table code_presence)
	dislipidemia	Number	dislipidemia (see table code_presence)
	claudicatio_intermittens	Number	claudicatio_intermittens (see table code_presence)
	stroke_pregressi	Number	previous_stroke (see table code_presence)
	attacchi_ischemici_transitori	Number	(TIA) transient_ischemic_attacks (see table code_presence)
	fumatore	Number	smoker (see table code_presence)
	consumo_alcoolici	Number	drinker(alcool) (see table code_presence)
	contraccettivi_orali	Number	oral_contraceptive (see table code_presence)
	terapia_ormonale_sostitutiva	Number	hormone substitution terapy (see table code_presence)

Figure M.3: Structure of Data_anamnesis table

Data_anamnesis_therapies

This table stores details about the previous therapies and drugs given to the patient. The total number of records in this table is 784.

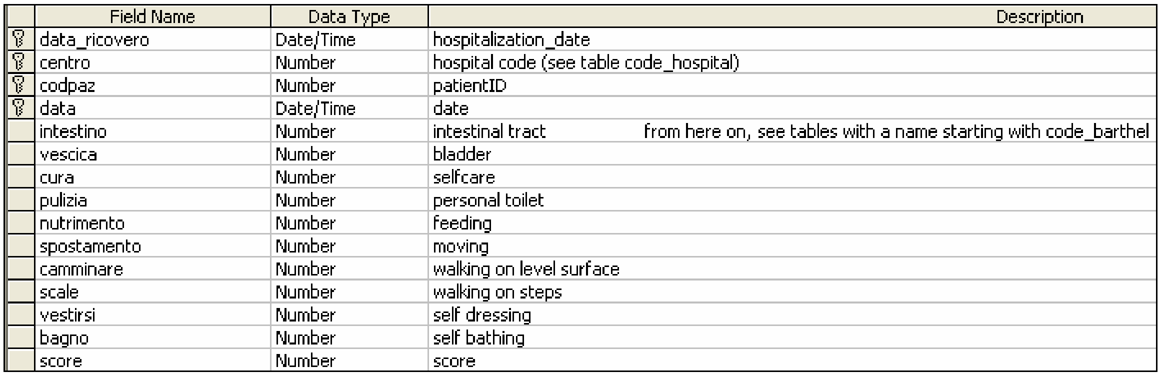


	Field Name	Data Type	Description
🔑	codpaz	Number	patientID
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
🔑	tipo_terapia	Number	therapy_type (see table code_therapies)
🔑	farmaco	Text	drug (see table code_drugs)
	dose	Number	dose
	unita_dose	Number	dose_unit (see table code_dose_unit)
	quante_volte	Number	how many times
	unita_tempo	Number	time_unit (see table code_time_unit) (example of dose + dose_unit + times + time_unit = 1 tablet 3 times a day)

Figure M.4: Structure of Data_anamnesis_therapies table

Data_barthel

This table stores details about the level of patient's disability. The total number of records in this table is 1240.

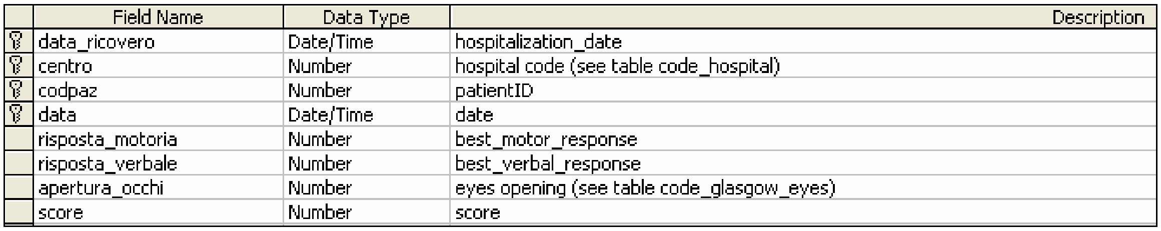


	Field Name	Data Type	Description
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
🔑	codpaz	Number	patientID
🔑	data	Date/Time	date
	intestino	Number	intestinal tract from here on, see tables with a name starting with code_barthel
	vescica	Number	bladder
	cura	Number	selfcare
	pulizia	Number	personal toilet
	nutrimento	Number	feeding
	spostamento	Number	moving
	camminare	Number	walking on level surface
	scale	Number	walking on steps
	vestirsi	Number	self dressing
	bagno	Number	self bathing
	score	Number	score

Figure M.5: Structure of Data_barthel table

Data_Glasgow_coma_scale

This table stores details about the patient's level of consciousness. The total number of records in this table is 233.

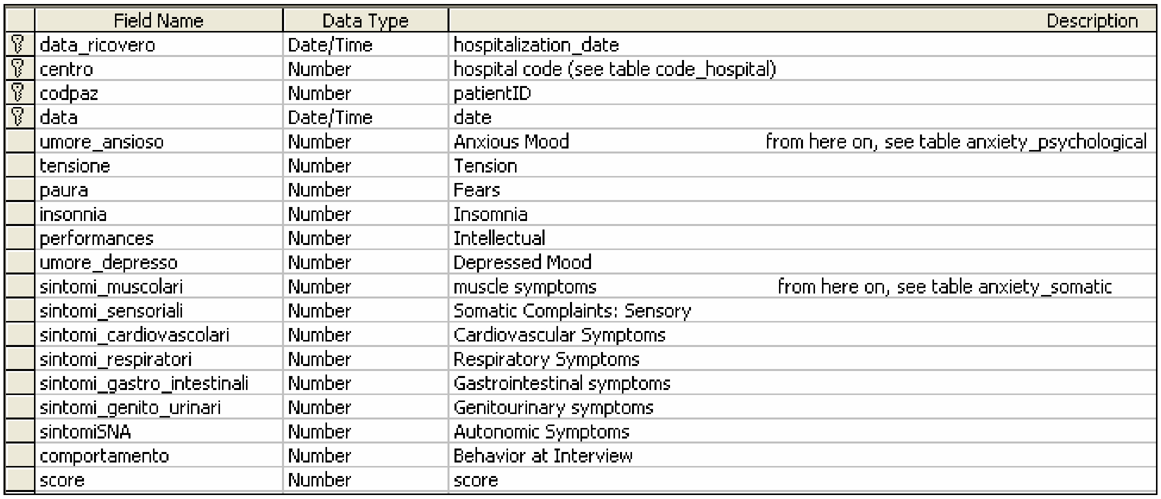


	Field Name	Data Type	Description
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
🔑	codpaz	Number	patientID
🔑	data	Date/Time	date
	risposta_motoria	Number	best_motor_response
	risposta_verbale	Number	best_verbal_response
	apertura_occhi	Number	eyes opening (see table code_glasgow_eyes)
	score	Number	score

Figure M.6: Structure of Data_Glasgow_coma_scale table

Data_Hamilton_anxiety

This table records anxiety condition of patients. For example, his degree of fear, tension, insomnia etc. are stored. The total number of records in this table is 592.

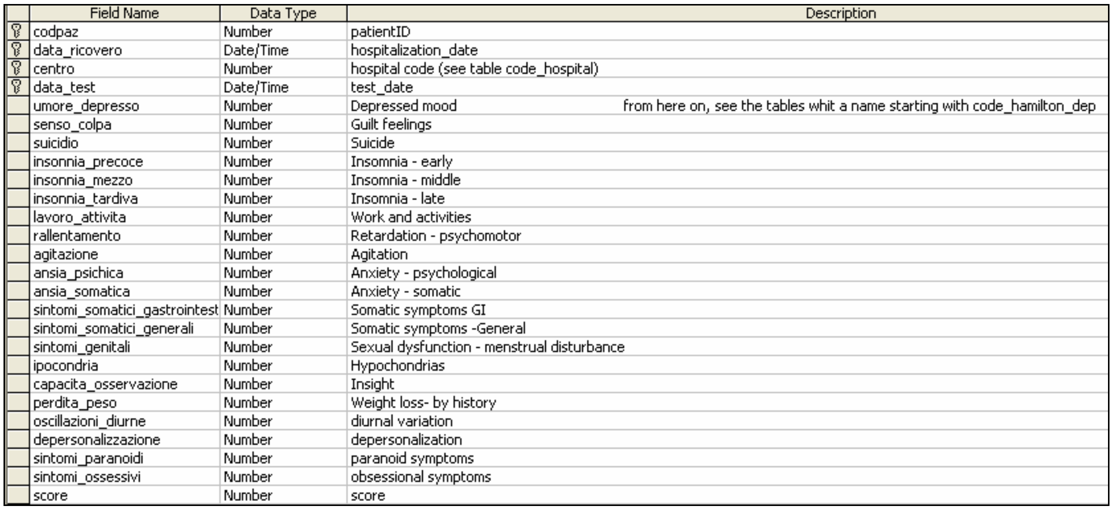


	Field Name	Data Type	Description
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
🔑	codpaz	Number	patientID
🔑	data	Date/Time	date
	umore_ansioso	Number	Anxious Mood from here on, see table anxiety_psychological
	tensione	Number	Tension
	paura	Number	Fears
	insonnia	Number	Insomnia
	performances	Number	Intellectual
	umore_depresso	Number	Depressed Mood
	sintomi_muscolari	Number	muscle symptoms from here on, see table anxiety_somatic
	sintomi_sensoriali	Number	Somatic Complaints: Sensory
	sintomi_cardiovascolari	Number	Cardiovascular Symptoms
	sintomi_respiratori	Number	Respiratory Symptoms
	sintomi_gastro_intestinali	Number	Gastrointestinal symptoms
	sintomi_genito_urinari	Number	Genitourinary symptoms
	sintomiSNA	Number	Autonomic Symptoms
	comportamento	Number	Behavior at Interview
	score	Number	score

Figure M.7: Structure of Data_Hamilton_anxiety table

Data_Hamilton_depression

This table stores details about any patient's state during the sub-acute phase. This table uses pre-defined codes for most of the parameters that define the state of any patient during sub-acute phase, therefore this table refers to many other tables (identified by *code_* , but we do not show these tables). The total number of records in this table is 590.

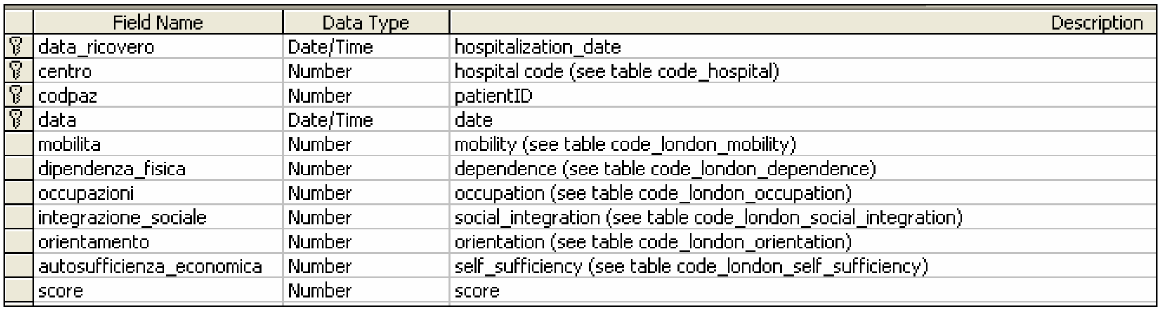


	Field Name	Data Type	Description
🔑	codpaz	Number	patientID
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
🔑	data_test	Date/Time	test_date
	umore_depresso	Number	Depressed mood from here on, see the tables whit a name starting with code_hamilton_dep
	senso_colpa	Number	Guilt feelings
	suicidio	Number	Suicide
	insonnia_precoce	Number	Insomnia - early
	insonnia_mezzo	Number	Insomnia - middle
	insonnia_tardiva	Number	Insomnia - late
	lavoro_attivita	Number	Work and activities
	rallentamento	Number	Retardation - psychomotor
	agitazione	Number	Agitation
	ansia_psichica	Number	Anxiety - psychological
	ansia_somatica	Number	Anxiety - somatic
	sintomi_somatici_gastrointest	Number	Somatic symptoms GI
	sintomi_somatici_generali	Number	Somatic symptoms -General
	sintomi_genitali	Number	Sexual dysfunction - menstrual disturbance
	ipocondria	Number	Hypochondrias
	capacita_osservazione	Number	Insight
	perdita_peso	Number	Weight loss- by history
	oscillazioni_diurne	Number	diurnal variation
	depersonalizzazione	Number	depersonalization
	sintomi_paranoidi	Number	paranoid symptoms
	sintomi_ossessivi	Number	obsessional symptoms
	score	Number	score

Figure M.8: Structure of Data_Hamilton_depression table

Data_London_handicap_scale

This table stores details about the extent of any patient's mental, physical and social handicap. The table also records the degree to which he is self dependent or dependent on others. The total number of records in this table is 660.

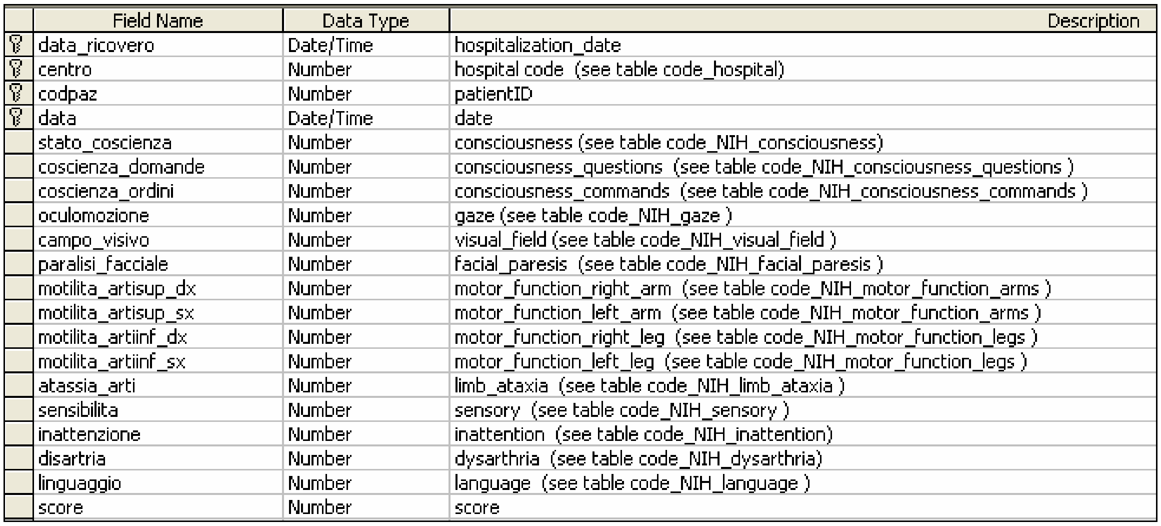


	Field Name	Data Type	Description
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
🔑	codpaz	Number	patientID
🔑	data	Date/Time	date
	mobilita	Number	mobility (see table code_london_mobility)
	dipendenza_fisica	Number	dependence (see table code_london_dependence)
	occupazioni	Number	occupation (see table code_london_occupation)
	integrazione_sociale	Number	social_integration (see table code_london_social_integration)
	orientamento	Number	orientation (see table code_london_orientation)
	autosufficienza_economica	Number	self_sufficiency (see table code_london_self_sufficiency)
	score	Number	score

Figure M.9: Structure of Data_London_handicap_scale table

Data_NIH

This table stores data about neurological deficit in the patient. The total number of records in this table is 1153.

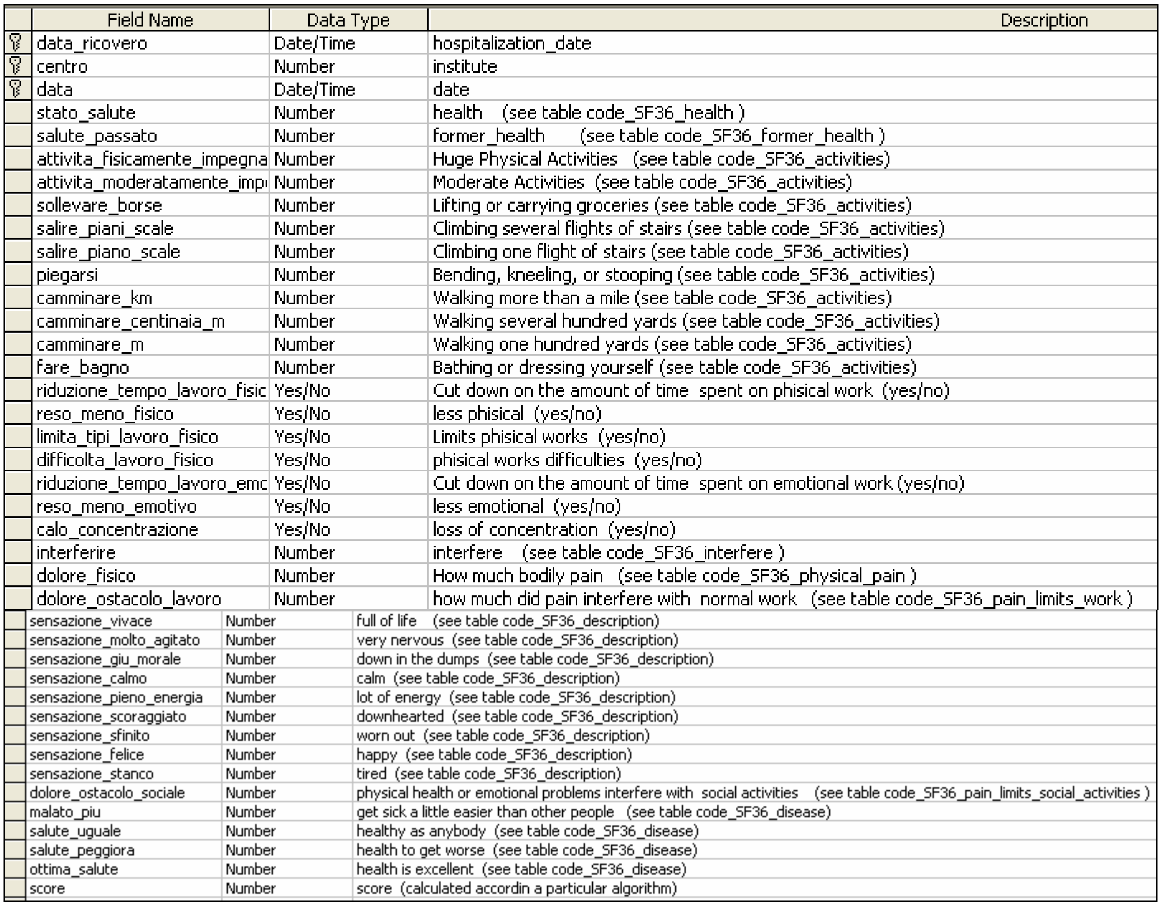


	Field Name	Data Type	Description
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
🔑	codpaz	Number	patientID
🔑	data	Date/Time	date
	stato_coscienza	Number	consciousness (see table code_NIH_consciousness)
	coscienza_domande	Number	consciousness_questions (see table code_NIH_consciousness_questions)
	coscienza_ordini	Number	consciousness_commands (see table code_NIH_consciousness_commands)
	oculomozione	Number	gaze (see table code_NIH_gaze)
	campo_visivo	Number	visual_field (see table code_NIH_visual_field)
	paralisi_facciale	Number	facial_paresis (see table code_NIH_facial_paresis)
	motilita_artisup_dx	Number	motor_function_right_arm (see table code_NIH_motor_function_arms)
	motilita_artisup_sx	Number	motor_function_left_arm (see table code_NIH_motor_function_arms)
	motilita_artiinf_dx	Number	motor_function_right_leg (see table code_NIH_motor_function_legs)
	motilita_artiinf_sx	Number	motor_function_left_leg (see table code_NIH_motor_function_legs)
	atassia_arti	Number	limb_ataxia (see table code_NIH_limb_ataxia)
	sensibilita	Number	sensory (see table code_NIH_sensory)
	inattenzione	Number	inattention (see table code_NIH_inattention)
	disartria	Number	dysarthria (see table code_NIH_dysarthria)
	linguaggio	Number	language (see table code_NIH_language)
	score	Number	score

Figure M.10: Structure of Data_NIH table

Data_SF36

This table stores details about the patient's state like physical strength, body pain, mental state (happy/sad) etc. The total number of records in this table is 502.

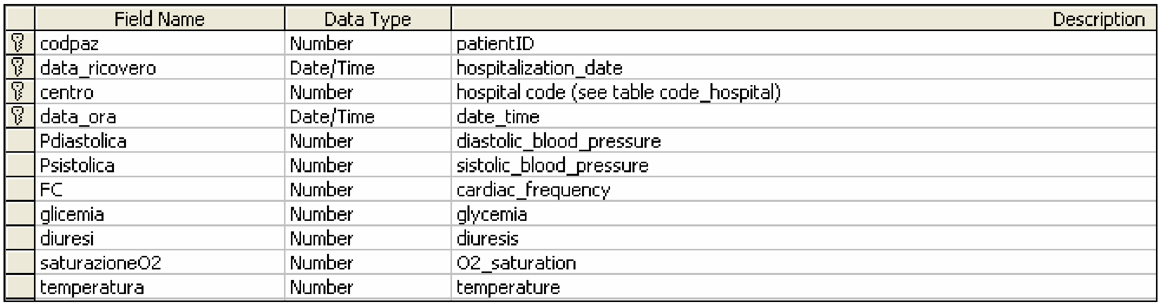


	Field Name	Data Type	Description
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	institute
🔑	data	Date/Time	date
	stato_salute	Number	health (see table code_SF36_health)
	salute_passato	Number	former_health (see table code_SF36_former_health)
	attivita_fisicamente_impegna	Number	Huge Physical Activities (see table code_SF36_activities)
	attivita_moderatamente_impı	Number	Moderate Activities (see table code_SF36_activities)
	sollevare_borse	Number	Lifting or carrying groceries (see table code_SF36_activities)
	salire_piani_scale	Number	Climbing several flights of stairs (see table code_SF36_activities)
	salire_piano_scale	Number	Climbing one flight of stairs (see table code_SF36_activities)
	piegarsi	Number	Bending, kneeling, or stooping (see table code_SF36_activities)
	camminare_km	Number	Walking more than a mile (see table code_SF36_activities)
	camminare_centinaia_m	Number	Walking several hundred yards (see table code_SF36_activities)
	camminare_m	Number	Walking one hundred yards (see table code_SF36_activities)
	fare_bagno	Number	Bathing or dressing yourself (see table code_SF36_activities)
	riduzione_tempo_lavoro_fisic	Yes/No	Cut down on the amount of time spent on phisical work (yes/no)
	reso_meno_fisico	Yes/No	less phisical (yes/no)
	limita_tipi_lavoro_fisico	Yes/No	Limits phisical works (yes/no)
	difficolta_lavoro_fisico	Yes/No	phisical works difficulties (yes/no)
	riduzione_tempo_lavoro_emc	Yes/No	Cut down on the amount of time spent on emotional work (yes/no)
	reso_meno_emotivo	Yes/No	less emotional (yes/no)
	calo_concentrazione	Yes/No	loss of concentration (yes/no)
	interferire	Number	interfere (see table code_SF36_interfere)
	dolore_fisico	Number	How much bodily pain (see table code_SF36_physical_pain)
	dolore_ostacolo_lavoro	Number	how much did pain interfere with normal work (see table code_SF36_pain_limits_work)
	sensazione_vivace	Number	full of life (see table code_SF36_description)
	sensazione_molto_agitato	Number	very nervous (see table code_SF36_description)
	sensazione_giu_morale	Number	down in the dumps (see table code_SF36_description)
	sensazione_calmo	Number	calm (see table code_SF36_description)
	sensazione_pieno_energia	Number	lot of energy (see table code_SF36_description)
	sensazione_scoraggiato	Number	downhearted (see table code_SF36_description)
	sensazione_sfinito	Number	worn out (see table code_SF36_description)
	sensazione_felice	Number	happy (see table code_SF36_description)
	sensazione_stanco	Number	tired (see table code_SF36_description)
	dolore_ostacolo_sociale	Number	physical health or emotional problems interfere with social activities (see table code_SF36_pain_limits_social_activities)
	malato_piu	Number	get sick a little easier than other people (see table code_SF36_disease)
	salute_uguale	Number	healthy as anybody (see table code_SF36_disease)
	salute_peggiora	Number	health to get worse (see table code_SF36_disease)
	ottima_salute	Number	health is excellent (see table code_SF36_disease)
	score	Number	score (calculated accordin a particular algorithm)

Figure M.11: Structure of Data_NSF36 table

Data_life_parameters

As the name suggests, the table stores data about various life parameters like body temperature, pulse rate etc. The total number of records in this table is 1563.

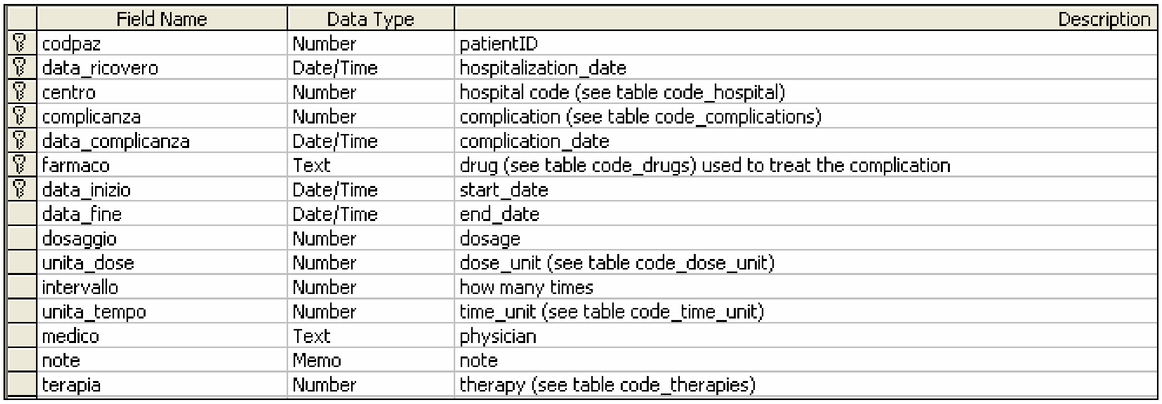


	Field Name	Data Type	Description
🔑	codpaz	Number	patientID
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
🔑	data_ora	Date/Time	date_time
	Pdiastolica	Number	diastolic_blood_pressure
	Psistolica	Number	sistolic_blood_pressure
	FC	Number	cardiac_frequency
	glicemia	Number	glycemia
	diuresi	Number	diuresis
	saturazioneO2	Number	O2_saturation
	temperatura	Number	temperature

Figure M.12: Structure of Data_life_parameters table

Data_medical_complications

This table stores data about the complication a patient is suffering from and what drugs, and in which quantity the drugs are given to him. The total number of records in this table is 823.

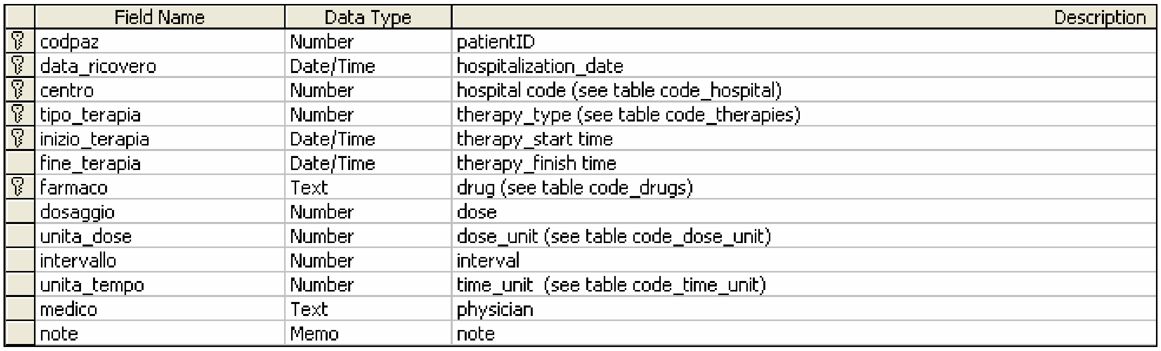


	Field Name	Data Type	Description
🔑	codpaz	Number	patientID
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
🔑	complicanza	Number	complication (see table code_complications)
🔑	data_complicanza	Date/Time	complication_date
🔑	farmaco	Text	drug (see table code_drugs) used to treat the complication
🔑	data_inizio	Date/Time	start_date
	data_fine	Date/Time	end_date
	dosaggio	Number	dosage
	unita_dose	Number	dose_unit (see table code_dose_unit)
	intervallo	Number	how many times
	unita_tempo	Number	time_unit (see table code_time_unit)
	medico	Text	physician
	note	Memo	note
	terapia	Number	therapy (see table code_therapies)

Figure M.13: Structure of Data_medical_complications table

Data_medical_therapy_acute_phase

This table stores details medical therapies given to any patient's state during acute phase. The total number of records in this table is 397.

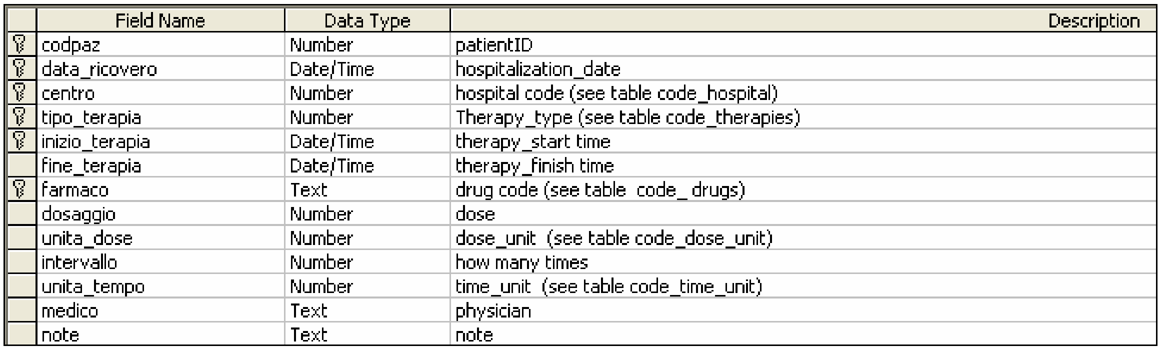


	Field Name	Data Type	Description
🔑	codpaz	Number	patientID
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
🔑	tipo_terapia	Number	therapy_type (see table code_therapies)
🔑	inizio_terapia	Date/Time	therapy_start time
	fine_terapia	Date/Time	therapy_finish time
🔑	farmaco	Text	drug (see table code_drugs)
	dosaggio	Number	dose
	unita_dose	Number	dose_unit (see table code_dose_unit)
	intervallo	Number	interval
	unita_tempo	Number	time_unit (see table code_time_unit)
	medico	Text	physician
	note	Memo	note

Figure M.14: Structure of Data_medical_therapy_acute_phase table

Data_medical_therapy_subacute_phase

This table stores details medical therapies given to any patient's state during subacute phase. The total number of records in this table is 1552.

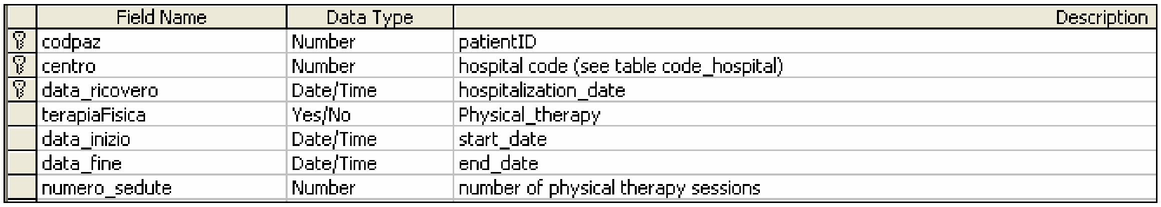


	Field Name	Data Type	Description
🔑	codpaz	Number	patientID
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
🔑	tipo_terapia	Number	Therapy_type (see table code_therapies)
🔑	inizio_terapia	Date/Time	therapy_start time
	fine_terapia	Date/Time	therapy_finish time
🔑	farmaco	Text	drug code (see table code_ drugs)
	dosaggio	Number	dose
	unita_dose	Number	dose_unit (see table code_dose_unit)
	intervallo	Number	how many times
	unita_tempo	Number	time_unit (see table code_time_unit)
	medico	Text	physician
	note	Text	note

Figure M.15: Structure of Data_medical_therapy_subacute_phase table

Data_physical_therapy

This table stores details physical therapy given to patients. The total number of records in this table is 143.

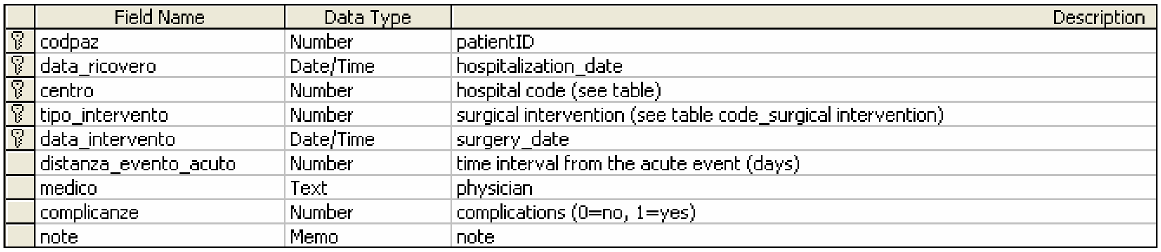


	Field Name	Data Type	Description
🔑	codpaz	Number	patientID
🔑	centro	Number	hospital code (see table code_hospital)
🔑	data_ricovero	Date/Time	hospitalization_date
	terapiaFisica	Yes/No	Physical_therapy
	data_inizio	Date/Time	start_date
	data_fine	Date/Time	end_date
	numero_sedute	Number	number of physical therapy sessions

Figure M.16: Structure of Data_physical_therapy table

Data_surgical_therapies

This table stores details surgical therapy given to patients. The total number of records in this table is 133.

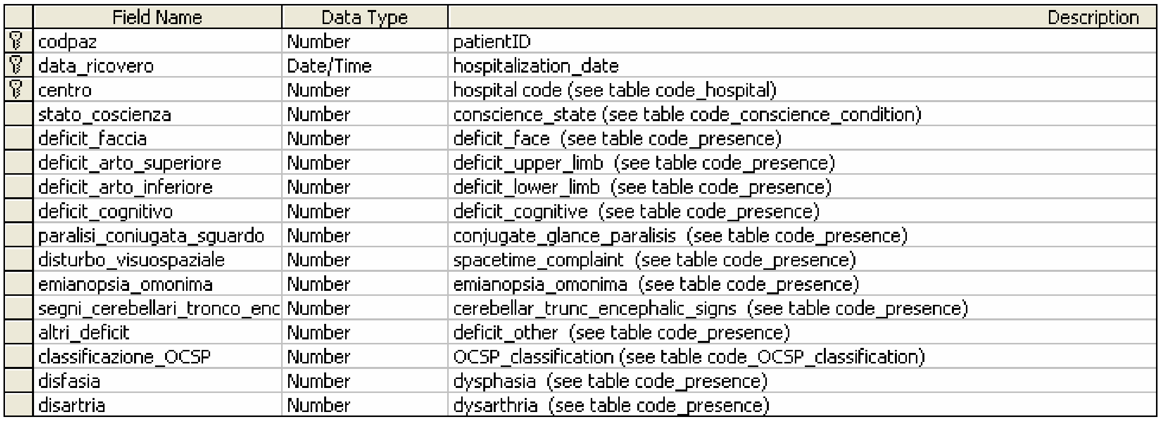


	Field Name	Data Type	Description
🔑	codpaz	Number	patientID
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table)
🔑	tipo_intervento	Number	surgical intervention (see table code_surgical intervention)
🔑	data_intervento	Date/Time	surgery_date
	distanza_evento_acuto	Number	time interval from the acute event (days)
	medico	Text	physician
	complicanze	Number	complications (0=no, 1=yes)
	note	Memo	note

Figure M.17: Structure of Data_surgical_therapies table

Data_neurological_examination

Both the neurological and objective examinations are done as soon as possible. 15-20 minutes from admission is the maximum time allowed to collect these data to prepare the treatment as quick as possible.The total number of records in this table is 384.

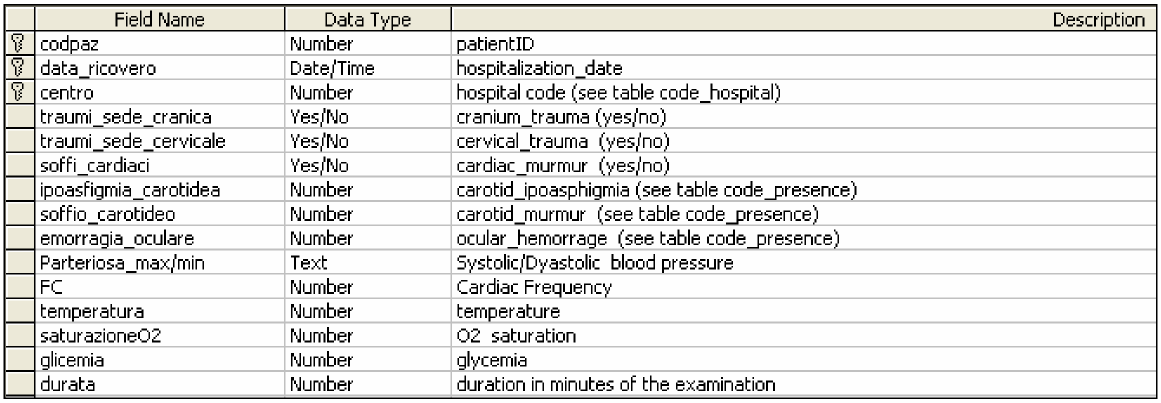


	Field Name	Data Type	Description
🔑	codpaz	Number	patientID
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
	stato_coscienza	Number	conscience_state (see table code_conscience_condition)
	deficit_faccia	Number	deficit_face (see table code_presence)
	deficit_arto_superiore	Number	deficit_upper_limb (see table code_presence)
	deficit_arto_inferiore	Number	deficit_lower_limb (see table code_presence)
	deficit_cognitivo	Number	deficit_cognitive (see table code_presence)
	paralisi_coniugata_sguardo	Number	conjugate_glance_paralisis (see table code_presence)
	disturbo_visuospaziale	Number	spacetime_complaint (see table code_presence)
	emianopsia_omonima	Number	emianopsia_omonima (see table code_presence)
	segni_cerebellari_tronco_enc	Number	cerebellar_trunc_encephalic_signs (see table code_presence)
	altri_deficit	Number	deficit_other (see table code_presence)
	classificazione_OCSP	Number	OCSP_classification (see table code_OCSP_classification)
	disfasia	Number	dysphasia (see table code_presence)
	disartria	Number	dysarthria (see table code_presence)

Figure M.18: Structure of Data_neurological_examination table

Data_objective_examination

This table stores details about any patient's state during the sub-acute phase. This table uses pre-defined codes for most of the parameters that define the state of any patient during sub-acute phase, therefore this table refers to many other tables (identified by *code_* , but we do not show these tables). The total number of records in this table is 383.

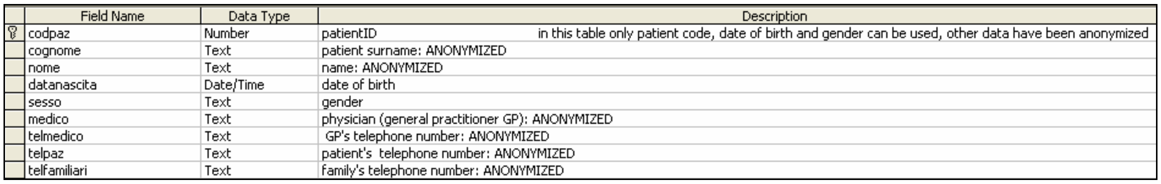


	Field Name	Data Type	Description
🔑	codpaz	Number	patientID
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
	traumi_sede_cranica	Yes/No	cranium_trauma (yes/no)
	traumi_sede_cervicale	Yes/No	cervical_trauma (yes/no)
	soffi_cardiaci	Yes/No	cardiac_murmur (yes/no)
	ipoasfigmia_carotidea	Number	carotid_ipoasphigmia (see table code_presence)
	soffio_carotideo	Number	carotid_murmur (see table code_presence)
	emorragia_oculare	Number	ocular_hemorrage (see table code_presence)
	Parteriosa_max/min	Text	Systolic/Dyastolic blood pressure
	FC	Number	Cardiac Frequency
	temperatura	Number	temperature
	saturazioneO2	Number	O2 saturation
	glicemia	Number	glycemia
	durata	Number	duration in minutes of the examination

Figure M.19: Structure of Data_objective_examination table

Data_personal_data

This table stores details personal details about patients: name, address, date of birth etc. The total number of records in this table is 386.

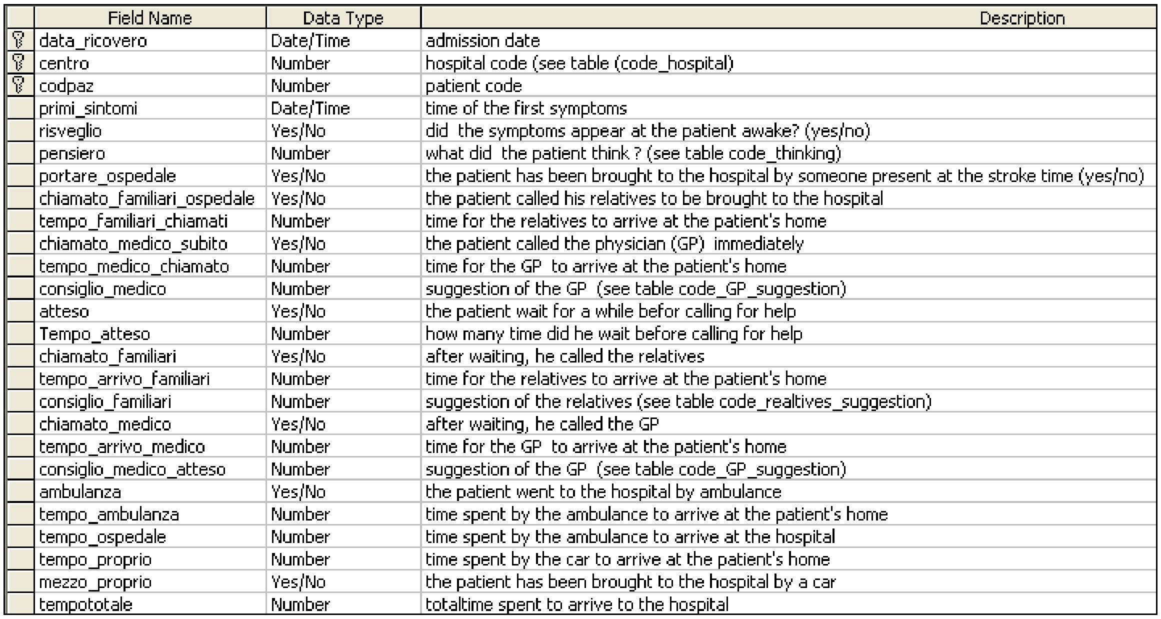


	Field Name	Data Type	Description	
🔑	codpaz	Number	patientID	in this table only patient code, date of birth and gender can be used, other data have been anonymized
	cognome	Text	patient surname: ANONYMIZED	
	nome	Text	name: ANONYMIZED	
	datanascita	Date/Time	date of birth	
	sesso	Text	gender	
	medico	Text	physician (general practitioner GP): ANONYMIZED	
	telmedico	Text	GP's telephone number: ANONYMIZED	
	telpaz	Text	patient's telephone number: ANONYMIZED	
	telfamiliari	Text	family's telephone number: ANONYMIZED	

Figure M.20: Structure of Data_personal_data table

Data_pre_hospital_phase

This table stores details about what happened when the patient has the first stroke symptoms. The total number of records in this table is 253.

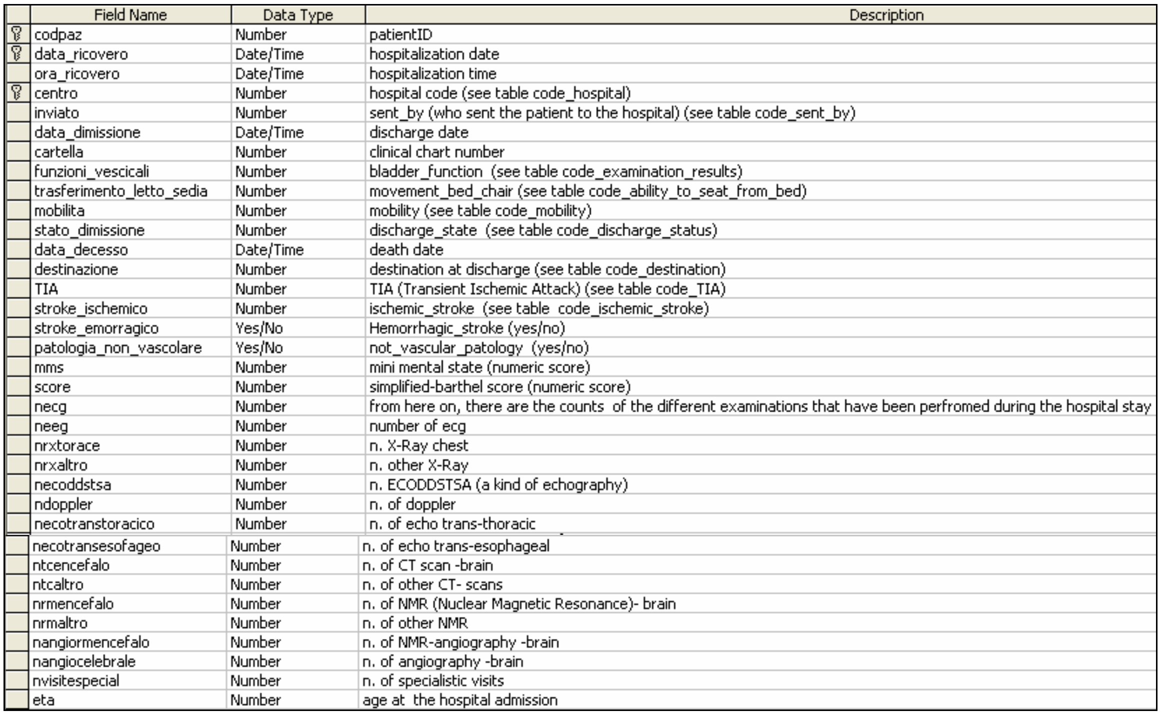


	Field Name	Data Type	Description
🔑	data_ricovero	Date/Time	admission date
🔑	centro	Number	hospital code (see table (code_hospital)
🔑	codpaz	Number	patient code
	primi_sintomi	Date/Time	time of the first symptoms
	risveglio	Yes/No	did the symptoms appear at the patient awake? (yes/no)
	pensiero	Number	what did the patient think ? (see table code_thinking)
	portare_ospedale	Yes/No	the patient has been brought to the hospital by someone present at the stroke time (yes/no)
	chiamato_familiari_ospedale	Yes/No	the patient called his relatives to be brought to the hospital
	tempo_familiari_chiamati	Number	time for the relatives to arrive at the patient's home
	chiamato_medico_subito	Yes/No	the patient called the physician (GP) immediately
	tempo_medico_chiamato	Number	time for the GP to arrive at the patient's home
	consiglio_medico	Number	suggestion of the GP (see table code_GP_suggestion)
	atteso	Yes/No	the patient wait for a while befor calling for help
	Tempo_atteso	Number	how many time did he wait before calling for help
	chiamato_familiari	Yes/No	after waiting, he called the relatives
	tempo_arrivo_familiari	Number	time for the relatives to arrive at the patient's home
	consiglio_familiari	Number	suggestion of the relatives (see table code_realtives_suggestion)
	chiamato_medico	Yes/No	after waiting, he called the GP
	tempo_arrivo_medico	Number	time for the GP to arrive at the patient's home
	consiglio_medico_atteso	Number	suggestion of the GP (see table code_GP_suggestion)
	ambulanza	Yes/No	the patient went to the hospital by ambulance
	tempo_ambulanza	Number	time spent by the ambulance to arrive at the patient's home
	tempo_ospedale	Number	time spent by the ambulance to arrive at the hospital
	tempo_proprio	Number	time spent by the car to arrive at the patient's home
	mezzo_proprio	Yes/No	the patient has been brought to the hospital by a car
	tempototale	Number	totaltime spent to arrive to the hospital

Figure M.21: Structure of Data_pre_hospital_phase table

Data_admission_discharge

This table stores details about patient's state during discharge, results of various tests done at discharge. The total number of records in this table is 386.

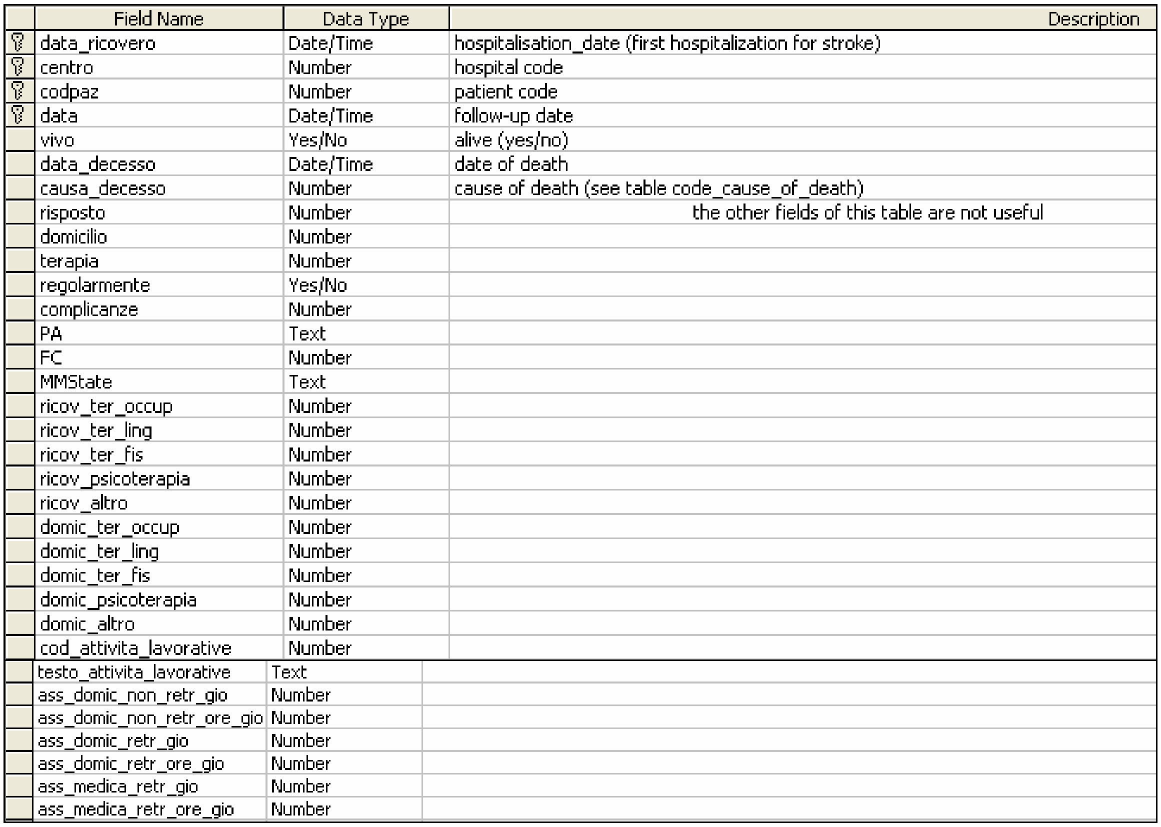


	Field Name	Data Type	Description
🔑	codpaz	Number	patientID
🔑	data_ricovero	Date/Time	hospitalization date
	ora_ricovero	Date/Time	hospitalization time
🔑	centro	Number	hospital code (see table code_hospital)
	inviato	Number	sent_by (who sent the patient to the hospital) (see table code_sent_by)
	data_dimissione	Date/Time	discharge date
	cartella	Number	clinical chart number
	funzioni_vescicali	Number	bladder_function (see table code_examination_results)
	trasferimento_letto_sedia	Number	movement_bed_chair (see table code_ability_to_seat_from_bed)
	mobilita	Number	mobility (see table code_mobility)
	stato_dimissione	Number	discharge_state (see table code_discharge_status)
	data_decesso	Date/Time	death date
	destinazione	Number	destination at discharge (see table code_destination)
	TIA	Number	TIA (Transient Ischemic Attack) (see table code_TIA)
	stroke_ischemico	Number	ischemic_stroke (see table code_ischemic_stroke)
	stroke_emorragico	Yes/No	Hemorrhagic_stroke (yes/no)
	patologia_non_vascolare	Yes/No	not_vascular_patology (yes/no)
	mms	Number	mini mental state (numeric score)
	score	Number	simplified-barthel score (numeric score)
	necg	Number	from here on, there are the counts of the different examinations that have been perfromed during the hospital stay
	neeg	Number	number of ecg
	nrxtorace	Number	n. X-Ray chest
	nrxaltro	Number	n. other X-Ray
	necoddstsa	Number	n. ECODDSTSA (a kind of echography)
	ndoppler	Number	n. of doppler
	necotranstoracico	Number	n. of echo trans-thoracic
	necotransesofageo	Number	n. of echo trans-esophageal
	ntcencefalo	Number	n. of CT scan -brain
	ntcaltro	Number	n. of other CT- scans
	nrmencefalo	Number	n. of NMR (Nuclear Magnetic Resonance)- brain
	nrmaltro	Number	n. of other NMR
	nangiormencefalo	Number	n. of NMR-angiography -brain
	nangiocelebrale	Number	n. of angiography -brain
	nvisitespecial	Number	n. of specialistic visits
	eta	Number	age at the hospital admission

Figure M.22: Structure of Data_admission_discharge table

Data_follow_up

This table stores details about the patient after he/she has been discharged from the hospital. The total number of records in this table is 499.

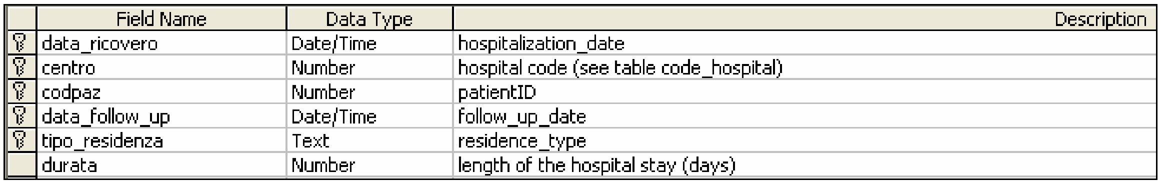


	Field Name	Data Type	Description
🔑	data_ricovero	Date/Time	hospitalisation_date (first hospitalization for stroke)
🔑	centro	Number	hospital code
🔑	codpaz	Number	patient code
🔑	data	Date/Time	follow-up date
	vivo	Yes/No	alive (yes/no)
	data_decesso	Date/Time	date of death
	causa_decesso	Number	cause of death (see table code_cause_of_death)
	risposto	Number	the other fields of this table are not useful
	domicilio	Number	
	terapia	Number	
	regolarmente	Yes/No	
	complicanze	Number	
	PA	Text	
	FC	Number	
	MMState	Text	
	ricov_ter_occup	Number	
	ricov_ter_ling	Number	
	ricov_ter_fis	Number	
	ricov_psicoterapia	Number	
	ricov_altro	Number	
	domic_ter_occup	Number	
	domic_ter_ling	Number	
	domic_ter_fis	Number	
	domic_psicoterapia	Number	
	domic_altro	Number	
	cod_attivita_lavorative	Number	
	testo_attivita_lavorative	Text	
	ass_domic_non_retr_gio	Number	
	ass_domic_non_retr_ore_gio	Number	
	ass_domic_retr_gio	Number	
	ass_domic_retr_ore_gio	Number	
	ass_medica_retr_gio	Number	
	ass_medica_retr_ore_gio	Number	

Figure M.23: Structure of Data_follow_up table

Data_re-hospitalization_during_follow_up

This table stores details about the patient when he is hospitalized during the follow up period. The total number of records in this table is 181.

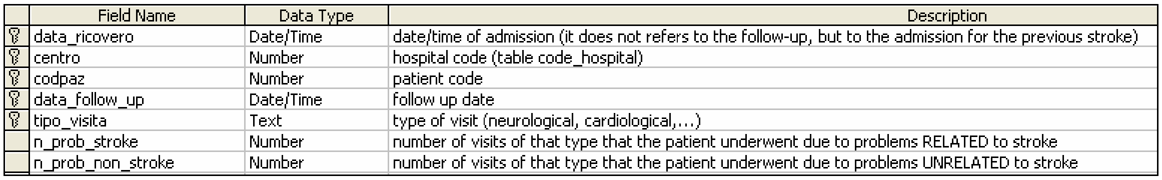


	Field Name	Data Type	Description
🔑	data_ricovero	Date/Time	hospitalization_date
🔑	centro	Number	hospital code (see table code_hospital)
🔑	codpaz	Number	patientID
🔑	data_follow_up	Date/Time	follow_up_date
🔑	tipo_residenza	Text	residence_type
	durata	Number	length of the hospital stay (days)

Figure M.24: Structure of Data_re-hospitalization_during_follow_up table

Data_visits_during_follow_up

This table stores details about the patient's visits to the hospital during the follow up period. The total number of records in this table is 248.



	Field Name	Data Type	Description
🔑	data_ricovero	Date/Time	date/time of admission (it does not refers to the follow-up, but to the admission for the previous stroke)
🔑	centro	Number	hospital code (table code_hospital)
🔑	codpaz	Number	patient code
🔑	data_follow_up	Date/Time	follow up date
🔑	tipo_visita	Text	type of visit (neurological, cardiological,…)
	n_prob_stroke	Number	number of visits of that type that the patient underwent due to problems RELATED to stroke
	n_prob_non_stroke	Number	number of visits of that type that the patient underwent due to problems UNRELATED to stroke

Figure M.25: Structure of Data_visits_during_follow_up table

N: MXML logs from Case study2

In this appendix the MXML log fragments for the logs used for experiments in Chapter 7 are given. For each event log, one process instance including its data attributes, audit trail entries, and other information is represented. It should be noted that the structure of the log is consistent with the DTD shown in Appendix A.

1. Measurements log

Measurements log consists of information about the various measurements done on the stroke patients. The PI information in this log is obtained from tables: *data_personal_data*, *data_admission_discharge, data_anamnesis, data_objective_examination, data_neurological_examination* and *data_subacute_*phase. The tables that provide information specific to the different measurements and conditions of the patients during these tests are: *barthel, glasgow_coma_scale, hamilton_anxiety, hamilton_depression, london_handicap_scale, NIH,* and *SF36*. These tables were combined to construct a log for measurements. The total number of cases in this log is 373. A part of this log is shown in Figure N.1.

```
<?xml version="1.0" encoding="UTF-8" ?>
<WorkflowLog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://is.tm.tue.nl/research/processmining/WorkflowLog.xsd" description="This log is
converted from the tables 'Process_Instances and Audit_Trail_Entries2 and Data_Attributes_Process_Instances and
Data_Attributes_Audit_Trail_Entries2' at the database 'jdbc:odbc:stroke20thmarch'">
  <Data>
    <Attribute name="app.name">ProM Import Framework</Attribute>
    <Attribute name="app.version">4.0 (Surfs Up)</Attribute>
    <Attribute name="java.vendor">Sun Microsystems Inc.</Attribute>
    <Attribute name="java.version">1.5.0_08</Attribute>
    <Attribute name="mxml.creator">MXMLib (http://promimport.sf.net/)</Attribute>
    <Attribute name="mxml.version">1.1</Attribute>
    <Attribute name="os.arch">x86</Attribute>
    <Attribute name="os.name">Windows XP</Attribute>
    <Attribute name="os.version">5.1</Attribute>
    <Attribute name="user.name">s041914</Attribute>
  </Data>
  <Source program="MsAccessDB"/>
  <Process id="GLOBAL" description="This log is converted from the tables 'Process_Instances and Audit_Trail_Entries2 and
Data_Attributes_Process_Instances and Data_Attributes_Audit_Trail_Entries2' at the database 'jdbc:odbc:stroke20thmarch'">
    <ProcessInstance id="111">
      <Data>
        <Attribute name="Hemorrhagic_stroke">False</Attribute>
        <Attribute name="TIA">0</Attribute>
        <Attribute name="age_hospital_admission">57.64384</Attribute>
        <Attribute name="allucinations">False</Attribute>
        <Attribute name="anamnesis_therapy_dose">0</Attribute>
        <Attribute name="anamnesis_therapy_dose_unit">0</Attribute>
        <Attribute name="anamnesis_therapy_drug">0</Attribute>
        <Attribute name="anamnesis_therapy_how_many_times">0</Attribute>
        <Attribute name="anamnesis_therapy_therapy_type">0</Attribute>
        <Attribute name="anamnesis_therapy_time_unit">0</Attribute>
        <Attribute name="angina">1</Attribute>
        <Attribute name="anxiety_panic">False</Attribute>
        <Attribute name="arterial_ipertension">1</Attribute>
        <Attribute name="ataxia">False</Attribute>
        <Attribute name="atrial_fibrillation">1</Attribute>
        <Attribute name="bilateral_inferior_limbs_deficit">False</Attribute>
        <Attribute name="bladder_function">3</Attribute>
        <Attribute name="blindness_monocular_temporary">False</Attribute>
        <Attribute name="cardiac_insufficiency">1</Attribute>
        <Attribute name="cause_death">0</Attribute>
        <Attribute name="cephalalgia">True</Attribute>
        <Attribute name="claudicatio_intermittens">1</Attribute>
        <Attribute name="destination_discharge">1</Attribute>
        <Attribute name="diabetes_mellitus">1</Attribute>
        <Attribute name="diagnostic_hypothesis">2</Attribute>
        <Attribute name="diplopia">False</Attribute>
        <Attribute name="discharge_date">30-04-1998</Attribute>
        <Attribute name="discharge_state">1</Attribute>
```

```
<Attribute name="dislipidemia">1</Attribute>
<Attribute name="drinker_alcohol_">2</Attribute>
<Attribute name="duration_preliminary_examination">5</Attribute>
<Attribute name="dysarthria">False</Attribute>
<Attribute name="dysphagia">False</Attribute>
<Attribute name="dysphasia">False</Attribute>
<Attribute name="emesis">False</Attribute>
<Attribute name="emianopsia_omonima">True</Attribute>
<Attribute name="gender">M</Attribute>
<Attribute name="hormone_substitution_terapy">0</Attribute>
<Attribute name="hospital_code">3</Attribute>
<Attribute name="hospitalization_date">24-04-1998</Attribute>
<Attribute name="ischemic_stroke">4</Attribute>
<Attribute name="loss_of_conscience">False</Attribute>
<Attribute name="mental_confusion">False</Attribute>
<Attribute name="mini_mental_state">30</Attribute>
<Attribute name="mobility">4</Attribute>
<Attribute name="movement_bed_chair">4</Attribute>
<Attribute name="myocardial_infarction">1</Attribute>
<Attribute name="neur_ex_OCSP_classification">5</Attribute>
<Attribute name="neur_ex_cerebellar_trunc_encephalic_signs">1</Attribute>
<Attribute name="neur_ex_conjugate_glance_paralisis">1</Attribute>
<Attribute name="neur_ex_conscience_state">1</Attribute>
<Attribute name="neur_ex_deficit_cognitive">1</Attribute>
<Attribute name="neur_ex_deficit_face">1</Attribute>
<Attribute name="neur_ex_deficit_lower_limb">1</Attribute>
<Attribute name="neur_ex_deficit_other">1</Attribute>
<Attribute name="neur_ex_deficit_upper_limb">1</Attribute>
<Attribute name="neur_ex_dysarthria">1</Attribute>
<Attribute name="neur_ex_dysphasia">1</Attribute>
<Attribute name="neur_ex_emianopsia_omonima">2</Attribute>
<Attribute name="neur_ex_spacetime_complaint">1</Attribute>
<Attribute name="neurological_signs_a">0</Attribute>
<Attribute name="neurological_signs_b">0</Attribute>
<Attribute name="not_vascular_patology">False</Attribute>
<Attribute name="number_CT_scan-brain">0</Attribute>
<Attribute name="number_ECODDSTSA">1</Attribute>
<Attribute name="number_NMR-angiography-brain">0</Attribute>
<Attribute name="number_NMR-brain">0</Attribute>
<Attribute name="number_X-Ray_chest">0</Attribute>
<Attribute name="number__eeg">0</Attribute>
<Attribute name="number_angiography-brain">0</Attribute>
<Attribute name="number_doppler">0</Attribute>
<Attribute name="number_echo_trans-esophageal">1</Attribute>
<Attribute name="number_echo_trans-thoracic">0</Attribute>
<Attribute name="number_of_ecg">1</Attribute>
<Attribute name="number_other_CT-_scans">0</Attribute>
<Attribute name="number_other_NMR">0</Attribute>
<Attribute name="number_other_X-Ray">0</Attribute>
<Attribute name="number_specialistic_visits">2</Attribute>
<Attribute name="obj_ex_Cardiac_Frequency">60</Attribute>
<Attribute name="obj_ex_O2saturation">0</Attribute>
<Attribute name="obj_ex_Systolic/Dyastolic__blood_pressure">160/90</Attribute>
<Attribute name="obj_ex_cardiac_murmur">False</Attribute>
<Attribute name="obj_ex_carotid_ipoasphigmia">0</Attribute>
<Attribute name="obj_ex_carotid_murmur">0</Attribute>
<Attribute name="obj_ex_cervical_trauma">False</Attribute>
<Attribute name="obj_ex_cranium_trauma">False</Attribute>
<Attribute name="obj_ex_duration">5</Attribute>
<Attribute name="obj_ex_glycemia">0</Attribute>
<Attribute name="obj_ex_ocular_hemorrage">0</Attribute>
<Attribute name="obj_ex_temperature">36</Attribute>
<Attribute name="oral_contraceptive">0</Attribute>
<Attribute name="patientAlive">yes</Attribute>
<Attribute name="previous_stroke">1</Attribute>
<Attribute name="rehospitalization">no</Attribute>
<Attribute name="seizure">False</Attribute>
<Attribute name="sensitive_cross-shaped_motor_disturbance">False</Attribute>
<Attribute name="sent_by">2</Attribute>
<Attribute name="simplified-barthel_score">19</Attribute>
```

```
    <Attribute name="smoker">2</Attribute>
    <Attribute name="space/time_disorientation">False</Attribute>
    <Attribute name="strength_deficit">False</Attribute>
    <Attribute name="stroke_side">1</Attribute>
    <Attribute name="subacute_Antithrombine">1</Attribute>
    <Attribute name="subacute_CT-encephalus">5</Attribute>
    <Attribute name="subacute_C_protein">1</Attribute>
    <Attribute name="subacute_C_protein_resistance">1</Attribute>
    <Attribute name="subacute_ESR">2</Attribute>
    <Attribute name="subacute_Echo_DDS_TSA">1</Attribute>
    <Attribute name="subacute_LAC">1</Attribute>
    <Attribute name="subacute_Magnetic_resonance_angiography">0</Attribute>
    <Attribute name="subacute_Magnetic_resonance_encephalus">0</Attribute>
    <Attribute name="subacute_PT">2</Attribute>
    <Attribute name="subacute_PTT">2</Attribute>
    <Attribute name="subacute_S_protein">1</Attribute>
    <Attribute name="subacute_TPHA">1</Attribute>
    <Attribute name="subacute_Transcranial_Dopple">0</Attribute>
    <Attribute name="subacute_VDRL">1</Attribute>
    <Attribute name="subacute_angiography">0</Attribute>
    <Attribute name="subacute_antiphospholipids_antibodies">1</Attribute>
    <Attribute name="subacute_cardiological_examination">False</Attribute>
    <Attribute name="subacute_complete_blood_count">2</Attribute>
    <Attribute name="subacute_glucose_load_curve">1</Attribute>
    <Attribute name="subacute_glycemia">2</Attribute>
    <Attribute name="subacute_hemoglobin_electrophoresis">1</Attribute>
    <Attribute name="subacute_homocysteine">1</Attribute>
    <Attribute name="subacute_neuropsychological_tests">0</Attribute>
    <Attribute name="subacute_serum_protein_electrophoresis">2</Attribute>
    <Attribute name="subacute_thransesophageal_echography">0</Attribute>
    <Attribute name="subacute_thransthorax_echocardiography">3</Attribute>
    <Attribute name="subacute_total_cholestero">3</Attribute>
    <Attribute name="subacute_urine_Homocistein">1</Attribute>
    <Attribute name="symptoms_duration">27</Attribute>
    <Attribute name="symptoms_onset_modality">1</Attribute>
    <Attribute name="timestamp_symptoms_stroke">23-04-1998</Attribute>
    <Attribute name="transient_ischemic_attacks">2</Attribute>
    <Attribute name="unilateral_sensitive_deficit">False</Attribute>
    <Attribute name="unintentional_movements">False</Attribute>
    <Attribute name="vertigo">False</Attribute>
  </Data>
  <AuditTrailEntry>
    <Data>
      <Attribute name="bladder">2</Attribute>
      <Attribute name="feeding">2</Attribute>
      <Attribute name="hospital_code">3</Attribute>
      <Attribute name="intestinal_tract">2</Attribute>
      <Attribute name="moving">3</Attribute>
      <Attribute name="personal_toilet">2</Attribute>
      <Attribute name="score">20</Attribute>
      <Attribute name="self_bathing">1</Attribute>
      <Attribute name="self_dressing">2</Attribute>
      <Attribute name="selfcare">1</Attribute>
      <Attribute name="typeTask">Measurement_barthel</Attribute>
      <Attribute name="walking_on_level_surface">3</Attribute>
      <Attribute name="walking_on_steps">2</Attribute>
    </Data>
    <WorkflowModelElement>Measurement_barthel</WorkflowModelElement>
    <EventType>complete</EventType>
    <Timestamp>1998-04-24T00:00:00.000+01:00</Timestamp>
    <Originator></Originator>
  </AuditTrailEntry>
  <AuditTrailEntry>
    <WorkflowModelElement>Measurement_NIH</WorkflowModelElement>
    <EventType>complete</EventType>
    <Timestamp>1998-04-24T00:00:00.000+01:00</Timestamp>
    <Originator></Originator>
  </AuditTrailEntry>
  <AuditTrailEntry>
    <Data>
```

```
        <Attribute name="bladder">2</Attribute>
        <Attribute name="feeding">2</Attribute>
        <Attribute name="hospital_code">3</Attribute>
        <Attribute name="intestinal_tract">2</Attribute>
        <Attribute name="moving">3</Attribute>
        <Attribute name="personal_toilet">2</Attribute>
        <Attribute name="score">20</Attribute>
        <Attribute name="self_bathing">1</Attribute>
        <Attribute name="self_dressing">2</Attribute>
        <Attribute name="selfcare">1</Attribute>
        <Attribute name="typeTask">Measurement_barthel</Attribute>
        <Attribute name="walking_on_level_surface">3</Attribute>
        <Attribute name="walking_on_steps">2</Attribute>
      </Data>
      <WorkflowModelElement>Measurement_barthel</WorkflowModelElement>
      <EventType>complete</EventType>
      <Timestamp>1998-04-29T00:00:00.000+01:00</Timestamp>
      <Originator></Originator>
    </AuditTrailEntry>
    <AuditTrailEntry>
      <Data>
        <Attribute name="dependence">1</Attribute>
        <Attribute name="hospital_code">3</Attribute>
        <Attribute name="mobility">1</Attribute>
        <Attribute name="occupation">1</Attribute>
        <Attribute name="orientation">1</Attribute>
        <Attribute name="score">6</Attribute>
        <Attribute name="self_sufficiency">1</Attribute>
        <Attribute name="social_integration">1</Attribute>
        <Attribute name="typeTask">Measurement_london</Attribute>
      </Data>
      <WorkflowModelElement>Measurement_london</WorkflowModelElement>
      <EventType>complete</EventType>
      <Timestamp>1998-04-29T00:00:00.000+01:00</Timestamp>
      <Originator></Originator>
    </AuditTrailEntry>
    <AuditTrailEntry>
      <Data>
        <Attribute name="Anxious_Mood">1</Attribute>
        <Attribute name="Autonomic_Symptoms">0</Attribute>
        <Attribute name="Behavior_at_Interview">0</Attribute>
        <Attribute name="Cardiovascular_Symptoms">0</Attribute>
        <Attribute name="Depressed_Mood">0</Attribute>
        <Attribute name="Fears">0</Attribute>
        <Attribute name="Gastrointestinal_symptoms">0</Attribute>
        <Attribute name="Genitourinary_symptoms">0</Attribute>
        <Attribute name="Insomnia">0</Attribute>
        <Attribute name="Intellectual">0</Attribute>
        <Attribute name="Respiratory_Symptoms">0</Attribute>
        <Attribute name="Somatic_Complaints:_Sensory">0</Attribute>
        <Attribute name="Tension">1</Attribute>
        <Attribute name="hospital_code">3</Attribute>
        <Attribute name="muscle_symptoms">0</Attribute>
        <Attribute name="score">1</Attribute>
        <Attribute name="typeTask">Measurement_hamilton_anxiety</Attribute>
      </Data>
      <WorkflowModelElement>Measurement_hamilton_anxiety</WorkflowModelElement>
      <EventType>complete</EventType>
      <Timestamp>1998-04-29T00:00:00.000+01:00</Timestamp>
      <Originator></Originator>
    </AuditTrailEntry>
    <AuditTrailEntry>
      <Data>
        <Attribute name="Agitation">0</Attribute>
        <Attribute name="Anxiety_psychological">0</Attribute>
        <Attribute name="Anxiety_somatic">0</Attribute>
        <Attribute name="Depressed_mood">0</Attribute>
        <Attribute name="Guilt_feelings">0</Attribute>
        <Attribute name="Hypochondrias">0</Attribute>
        <Attribute name="Insight">0</Attribute>
      <Attribute name="Insomnia_early">0</Attribute>
      <Attribute name="Insomnia_late">0</Attribute>
      <Attribute name="Insomnia_middle">0</Attribute>
      <Attribute name="Retardation_psychomotor">0</Attribute>
      <Attribute name="Sexual_dysfunction">0</Attribute>
      <Attribute name="Somatic_symptoms_GI">0</Attribute>
      <Attribute name="Somatic_symptoms_General">0</Attribute>
      <Attribute name="Suicide">0</Attribute>
      <Attribute name="Weight_loss">0</Attribute>
      <Attribute name="Work_and_activities">0</Attribute>
      <Attribute name="depersonalization">0</Attribute>
      <Attribute name="diurnal_variation">0</Attribute>
      <Attribute name="hospitalization_date">24/04/1998</Attribute>
      <Attribute name="obsessional_symptoms">0</Attribute>
      <Attribute name="score">0</Attribute>
      <Attribute name="sparanoid_symptoms">0</Attribute>
      <Attribute name="typeTask">Measurement_hamilton_depression</Attribute>
    </Data>
    <WorkflowModelElement>Measurement_hamilton_depression</WorkflowModelElement>
    <EventType>complete</EventType>
    <Timestamp>1998-04-29T00:00:00.000+01:00</Timestamp>
    <Originator></Originator>
  </AuditTrailEntry>
  <AuditTrailEntry>
    <Data>
      <Attribute name="Bathing_or_dressing_yoursel">3</Attribute>
      <Attribute name="Bending__kneeling__or_stooping">3</Attribute>
      <Attribute name="Climbing_one_flight_stairs">3</Attribute>
      <Attribute name="Climbing_several_flights_stair">3</Attribute>
      <Attribute name="Huge_Physical_Activities">3</Attribute>
      <Attribute name="Lifting_or_carrying_groceries">3</Attribute>
      <Attribute name="Limits_phisical_works">0</Attribute>
      <Attribute name="Moderate_Activities">3</Attribute>
      <Attribute name="calm">5</Attribute>
      <Attribute name="cut_down_amount_time_emotional_work">0</Attribute>
      <Attribute name="cut_down_amount_time_physical_work">0</Attribute>
      <Attribute name="down_in_the_dumps">6</Attribute>
      <Attribute name="downhearted">6</Attribute>
      <Attribute name="former_health">3</Attribute>
      <Attribute name="full_of_life">4</Attribute>
      <Attribute name="happy">5</Attribute>
      <Attribute name="health">3</Attribute>
      <Attribute name="health_is_excellent">2</Attribute>
      <Attribute name="health_to_get_worse">4</Attribute>
      <Attribute name="healthy_as_anybody">2</Attribute>
      <Attribute name="institute">3</Attribute>
      <Attribute name="interfere">1</Attribute>
      <Attribute name="less_emotional">0</Attribute>
      <Attribute name="less_phisical">0</Attribute>
      <Attribute name="loss_of_concentration">0</Attribute>
      <Attribute name="lot_of_energy">5</Attribute>
      <Attribute name="pain_interfere_normal_work">1</Attribute>
      <Attribute name="pain_interfere_social">5</Attribute>
      <Attribute name="physical_pain">1</Attribute>
      <Attribute name="physical_works_difficulties">0</Attribute>
      <Attribute name="score">103</Attribute>
      <Attribute name="sick_little_easier_compared_other_people">5</Attribute>
      <Attribute name="tired">5</Attribute>
      <Attribute name="typeTask">Measurement_SF36</Attribute>
      <Attribute name="very_nervous">5</Attribute>
      <Attribute name="walking_hundred_m">3</Attribute>
      <Attribute name="walking_km">3</Attribute>
      <Attribute name="walking_several_hundred_m">3</Attribute>
      <Attribute name="worn_out">5</Attribute>
    </Data>
    <WorkflowModelElement>Measurement_SF36</WorkflowModelElement>
    <EventType>complete</EventType>
    <Timestamp>1998-04-29T00:00:00.000+01:00</Timestamp>
    <Originator></Originator>
  </AuditTrailEntry>
```

```
    <AuditTrailEntry>
      <WorkflowModelElement>Measurement_NIH</WorkflowModelElement>
      <EventType>complete</EventType>
      <Timestamp>1998-04-29T00:00:00.000+01:00</Timestamp>
      <Originator></Originator>
    </AuditTrailEntry>
    <AuditTrailEntry>
      <Data>
        <Attribute name="bladder">2</Attribute>
        <Attribute name="feeding">2</Attribute>
        <Attribute name="hospital_code">3</Attribute>
        <Attribute name="intestinal_tract">2</Attribute>
        <Attribute name="moving">3</Attribute>
        <Attribute name="personal_toilet">2</Attribute>
        <Attribute name="score">20</Attribute>
        <Attribute name="self_bathing">1</Attribute>
        <Attribute name="self_dressing">2</Attribute>
        <Attribute name="selfcare">1</Attribute>
        <Attribute name="typeTask">Measurement_barthel</Attribute>
        <Attribute name="walking_on_level_surface">3</Attribute>
        <Attribute name="walking_on_steps">2</Attribute>
      </Data>
      <WorkflowModelElement>Measurement_barthel</WorkflowModelElement>
      <EventType>complete</EventType>
      <Timestamp>1998-09-22T00:00:00.000+01:00</Timestamp>
      <Originator></Originator>
    </AuditTrailEntry>
    <AuditTrailEntry>
      <Data>
        <Attribute name="dependence">1</Attribute>
        <Attribute name="hospital_code">3</Attribute>
        <Attribute name="mobility">1</Attribute>
        <Attribute name="occupation">1</Attribute>
        <Attribute name="orientation">1</Attribute>
        <Attribute name="score">6</Attribute>
        <Attribute name="self_sufficiency">1</Attribute>
        <Attribute name="social_integration">1</Attribute>
        <Attribute name="typeTask">Measurement_london</Attribute>
      </Data>
      <WorkflowModelElement>Measurement_london</WorkflowModelElement>
      <EventType>complete</EventType>
      <Timestamp>1998-09-22T00:00:00.000+01:00</Timestamp>
      <Originator></Originator>
    </AuditTrailEntry>
    <AuditTrailEntry>
      <Data>
        <Attribute name="Anxious_Mood">0</Attribute>
        <Attribute name="Autonomic_Symptoms">0</Attribute>
        <Attribute name="Behavior_at_Interview">0</Attribute>
        <Attribute name="Cardiovascular_Symptoms">0</Attribute>
        <Attribute name="Depressed_Mood">0</Attribute>
        <Attribute name="Fears">0</Attribute>
        <Attribute name="Gastrointestinal_symptoms">0</Attribute>
        <Attribute name="Genitourinary_symptoms">0</Attribute>
        <Attribute name="Insomnia">0</Attribute>
        <Attribute name="Intellectual">0</Attribute>
        <Attribute name="Respiratory_Symptoms">0</Attribute>
        <Attribute name="Somatic_Complaints:_Sensory">0</Attribute>
        <Attribute name="Tension">1</Attribute>
        <Attribute name="hospital_code">3</Attribute>
        <Attribute name="muscle_symptoms">0</Attribute>
        <Attribute name="score">1</Attribute>
        <Attribute name="typeTask">Measurement_hamilton_anxiety</Attribute>
      </Data>
      <WorkflowModelElement>Measurement_hamilton_anxiety</WorkflowModelElement>
      <EventType>complete</EventType>
      <Timestamp>1998-09-22T00:00:00.000+01:00</Timestamp>
      <Originator></Originator>
    </AuditTrailEntry>
    <AuditTrailEntry>
    <Data>
      <Attribute name="Agitation">0</Attribute>
      <Attribute name="Anxiety_psychological">0</Attribute>
      <Attribute name="Anxiety_somatic">0</Attribute>
      <Attribute name="Depressed_mood">0</Attribute>
      <Attribute name="Guilt_feelings">0</Attribute>
      <Attribute name="Hypochondrias">0</Attribute>
      <Attribute name="Insight">0</Attribute>
      <Attribute name="Insomnia_early">0</Attribute>
      <Attribute name="Insomnia_late">0</Attribute>
      <Attribute name="Insomnia_middle">0</Attribute>
      <Attribute name="Retardation_psychomotor">0</Attribute>
      <Attribute name="Sexual_dysfunction">0</Attribute>
      <Attribute name="Somatic_symptoms_GI">0</Attribute>
      <Attribute name="Somatic_symptoms_General">0</Attribute>
      <Attribute name="Suicide">0</Attribute>
      <Attribute name="Weight_loss">0</Attribute>
      <Attribute name="Work_and_activities">0</Attribute>
      <Attribute name="depersonalization">0</Attribute>
      <Attribute name="diurnal_variation">0</Attribute>
      <Attribute name="hospitalization_date">24/04/1998</Attribute>
      <Attribute name="obsessional_symptoms">0</Attribute>
      <Attribute name="score">0</Attribute>
      <Attribute name="sparanoid_symptoms">0</Attribute>
      <Attribute name="typeTask">Measurement_hamilton_depression</Attribute>
    </Data>
    <WorkflowModelElement>Measurement_hamilton_depression</WorkflowModelElement>
    <EventType>complete</EventType>
    <Timestamp>1998-09-22T00:00:00.000+01:00</Timestamp>
    <Originator></Originator>
  </AuditTrailEntry>
  <AuditTrailEntry>
    <Data>
      <Attribute name="Bathing_or_dressing_yoursel">3</Attribute>
      <Attribute name="Bending__kneeling__or_stooping">3</Attribute>
      <Attribute name="Climbing_one_flight_stairs">3</Attribute>
      <Attribute name="Climbing_several_flights_stair">3</Attribute>
      <Attribute name="Huge_Physical_Activities">3</Attribute>
      <Attribute name="Lifting_or_carrying_groceries">3</Attribute>
      <Attribute name="Limits_phisical_works">0</Attribute>
      <Attribute name="Moderate_Activities">3</Attribute>
      <Attribute name="calm">5</Attribute>
      <Attribute name="cut_down_amount_time_emotional_work">0</Attribute>
      <Attribute name="cut_down_amount_time_physical_work">0</Attribute>
      <Attribute name="down_in_the_dumps">6</Attribute>
      <Attribute name="downhearted">6</Attribute>
      <Attribute name="former_health">3</Attribute>
      <Attribute name="full_of_life">4</Attribute>
      <Attribute name="happy">5</Attribute>
      <Attribute name="health">3</Attribute>
      <Attribute name="health_is_excellent">2</Attribute>
      <Attribute name="health_to_get_worse">4</Attribute>
      <Attribute name="healthy_as_anybody">2</Attribute>
      <Attribute name="institute">3</Attribute>
      <Attribute name="interfere">1</Attribute>
      <Attribute name="less_emotional">0</Attribute>
      <Attribute name="less_phisical">0</Attribute>
      <Attribute name="loss_of_concentration">0</Attribute>
      <Attribute name="lot_of_energy">5</Attribute>
      <Attribute name="pain_interfere_normal_work">1</Attribute>
      <Attribute name="pain_interfere_social">5</Attribute>
      <Attribute name="physical_pain">1</Attribute>
      <Attribute name="physical_works_difficulties">0</Attribute>
      <Attribute name="score">103</Attribute>
      <Attribute name="sick_little_easier_compared_other_people">5</Attribute>
      <Attribute name="tired">5</Attribute>
      <Attribute name="typeTask">Measurement_SF36</Attribute>
      <Attribute name="very_nervous">5</Attribute>
      <Attribute name="walking_hundred_m">3</Attribute>
      <Attribute name="walking_km">3</Attribute>
```

```
        <Attribute name="walking_several_hundred_m">3</Attribute>
        <Attribute name="worn_out">5</Attribute>
      </Data>
      <WorkflowModelElement>Measurement_SF36</WorkflowModelElement>
      <EventType>complete</EventType>
      <Timestamp>1998-09-22T00:00:00.000+01:00</Timestamp>
      <Originator></Originator>
    </AuditTrailEntry>
    <AuditTrailEntry>
      <WorkflowModelElement>Measurement_NIH</WorkflowModelElement>
      <EventType>complete</EventType>
      <Timestamp>1998-09-22T00:00:00.000+01:00</Timestamp>
      <Originator></Originator>
    </AuditTrailEntry>
  </ProcessInstance>
```

Figure N.1: MXML log fragment for the *measurements* log

2. Therapies log

Therapies log consists of information about the various therapies that the stroke patients receive for certain medical complications. The PI information in this log is obtained from tables: *data_personal_data*, *data_admission_discharge, data_anamnesis, data_objective_examination, data_neurological_examination* and *data_subacute*_phase. The tables that provide information specific to the different therapies and medical complications are: *medical complications, medical_therapy_acute_phase, medical_therapy_subacute_phase, and physical* therapy. These tables were combined to construct a log for measurements. The total number of cases in this log is 380. A part of this log is shown in Figure N.2.

```
<?xml version="1.0" encoding="UTF-8"?>
<WorkflowLog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="WorkflowLog.xsd" description="Exported by ProM framework from This log is
converted from the tables 'Process_Instances and Audit_Trail_Entries and Data_Attributes_Process_Instances and
Data_Attributes_Audit_Trail_Entries' at the database 'jdbc:odbc:italian'">
<Data>
<Attribute name="mxml.version">1.0</Attribute>
<Attribute name="java.version">1.5.0_06</Attribute>
<Attribute name="user.name">s041914</Attribute>
<Attribute name="os.name">Windows XP</Attribute>
<Attribute name="os.arch">x86</Attribute>
<Attribute name="java.vendor">Sun Microsystems Inc.</Attribute>
<Attribute name="os.version">5.1</Attribute>
</Data>
<Source program="MS Access database">
<Data>
<Attribute name="program">MS Access database</Attribute>
</Data>
</Source>
<Process id="Process_Instances" description="A(n) MS Access database process.">
<ProcessInstance id="156" description="">
<Data>
<Attribute name="diplopia">False</Attribute>
<Attribute name="neurological_signs_a">1</Attribute>
<Attribute name="rehospitalization">no</Attribute>
<Attribute name="diabetes_mellitus">1</Attribute>
<Attribute name="loss_of_conscience">False</Attribute>
<Attribute name="neurological_signs_b">1</Attribute>
<Attribute name="bladder_function">3</Attribute>
<Attribute name="number_NMR-brain">0</Attribute>
<Attribute name="discharge_state">1</Attribute>
<Attribute name="number_echo_trans-thoracic">0</Attribute>
<Attribute name="number_of_ecg">1</Attribute>
<Attribute name="mini_mental_state">26</Attribute>
<Attribute name="transient_ischemic_attacks">1</Attribute>
<Attribute name="number__eeg">0</Attribute>
<Attribute name="TIA">0</Attribute>
<Attribute name="emesis">False</Attribute>
<Attribute name="cause_death">0</Attribute>
```

```
<Attribute name="number_angiography-brain">0</Attribute>
<Attribute name="drinker_alcohol_">1</Attribute>
<Attribute name="discharge_date">19-01-1998</Attribute>
<Attribute name="not_vascular_patology">False</Attribute>
<Attribute name="gender">F</Attribute>
<Attribute name="emianopsia_omonima">False</Attribute>
<Attribute name="hospital_code">1</Attribute>
<Attribute name="oral_contraceptive">0</Attribute>
<Attribute name="dysarthria">True</Attribute>
<Attribute name="ischemic_stroke">3</Attribute>
<Attribute name="seizure">False</Attribute>
<Attribute name="vertigo">False</Attribute>
<Attribute name="mental_confusion">True</Attribute>
<Attribute name="duration_preliminary_examination">10</Attribute>
<Attribute name="arterial_ipertension">2</Attribute>
<Attribute name="sent_by">4</Attribute>
<Attribute name="number_CT_scan-brain">0</Attribute>
<Attribute name="number_ECODDSTSA">1</Attribute>
<Attribute name="movement_bed_chair">4</Attribute>
<Attribute name="unintentional_movements">False</Attribute>
<Attribute name="hospitalization_date">07-01-1998</Attribute>
<Attribute name="anxiety_panic">False</Attribute>
<Attribute name="sensitive_cross-shaped_motor_disturbance">False</Attribute>
<Attribute name="smoker">1</Attribute>
<Attribute name="anamnesis_therapy_dose_unit">1</Attribute>
<Attribute name="symptoms_onset_modality">1</Attribute>
<Attribute name="number_doppler">0</Attribute>
<Attribute name="number_echo_trans-esophageal">1</Attribute>
<Attribute name="mobility">3</Attribute>
<Attribute name="dysphagia">False</Attribute>
<Attribute name="blindness_monocular_temporary">False</Attribute>
<Attribute name="anamnesis_therapy_drug">025682028</Attribute>
<Attribute name="cephalalgia">False</Attribute>
<Attribute name="space/time_disorientation">False</Attribute>
<Attribute name="patientAlive">yes</Attribute>
<Attribute name="number_other_X-Ray">0</Attribute>
<Attribute name="anamnesis_therapy_dose">1</Attribute>
<Attribute name="angina">1</Attribute>
<Attribute name="stroke_side">2</Attribute>
<Attribute name="anamnesis_therapy_time_unit">1</Attribute>
<Attribute name="destination_discharge">2</Attribute>
<Attribute name="allucinations">False</Attribute>
<Attribute name="atrial_fibrillation">1</Attribute>
<Attribute name="diagnostic_hypothesis">2</Attribute>
<Attribute name="previous_stroke">1</Attribute>
<Attribute name="simplified-barthel_score">17</Attribute>
<Attribute name="claudicatio_intermittens">1</Attribute>
<Attribute name="Hemorrhagic_stroke">False</Attribute>
<Attribute name="dislipidemia">1</Attribute>
<Attribute name="anamnesis_therapy_therapy_type">1</Attribute>
<Attribute name="age_hospital_admission">83,01096</Attribute>
<Attribute name="ataxia">False</Attribute>
<Attribute name="number_other_CT-_scans">0</Attribute>
<Attribute name="symptoms_duration">48</Attribute>
<Attribute name="number_other_NMR">0</Attribute>
<Attribute name="myocardial_infarction">1</Attribute>
<Attribute name="strength_deficit">True</Attribute>
<Attribute name="hormone_substitution_terapy">0</Attribute>
<Attribute name="anamnesis_therapy_how_many_times">0</Attribute>
<Attribute name="number_X-Ray_chest">0</Attribute>
<Attribute name="dysphasia">False</Attribute>
<Attribute name="cardiac_insufficiency">1</Attribute>
<Attribute name="number_NMR-angiography-brain">0</Attribute>
<Attribute name="number_specialistic_visits">0</Attribute>
<Attribute name="unilateral_sensitive_deficit">False</Attribute>
<Attribute name="bilateral_inferior_limbs_deficit">True</Attribute>
</Data>
<AuditTrailEntry>
<Data>
<Attribute name="Therapy_type">1</Attribute>
```

```
<Attribute name="drug">025682042</Attribute>
<Attribute name="time_unit">1</Attribute>
<Attribute name="typeTask">medical_therapy_subacute_phase</Attribute>
<Attribute name="hospital_code">1</Attribute>
<Attribute name="dose_unit">1</Attribute>
<Attribute name="dose">1</Attribute>
<Attribute name="how_many_times">1</Attribute>
</Data>
<WorkflowModelElement>therapyAcutePhase_type1</WorkflowModelElement>
<EventType>complete</EventType>
<Timestamp>1998-01-09T00:00:00.000+01:00</Timestamp>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="Therapy_type">20</Attribute>
<Attribute name="typeTask">medical_therapy_subacute_phase</Attribute>
<Attribute name="time_unit">1</Attribute>
<Attribute name="drug">030386054</Attribute>
<Attribute name="hospital_code">1</Attribute>
<Attribute name="dose_unit">3</Attribute>
<Attribute name="dose">1</Attribute>
<Attribute name="how_many_times">2</Attribute>
</Data>
<WorkflowModelElement>therapyAcutePhase_type20</WorkflowModelElement>
<EventType>complete</EventType>
<Timestamp>1998-01-13T00:00:00.000+01:00</Timestamp>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="Therapy_type">14</Attribute>
<Attribute name="drug">023075029</Attribute>
<Attribute name="time_unit">1</Attribute>
<Attribute name="typeTask">medical_therapy_subacute_phase</Attribute>
<Attribute name="hospital_code">1</Attribute>
<Attribute name="dose_unit">1</Attribute>
<Attribute name="dose">1</Attribute>
<Attribute name="how_many_times">1</Attribute>
</Data>
<WorkflowModelElement>therapyAcutePhase_type14</WorkflowModelElement>
<EventType>complete</EventType>
<Timestamp>1998-01-18T00:00:00.000+01:00</Timestamp>
</AuditTrailEntry>
<AuditTrailEntry>
<Data>
<Attribute name="typeTask">physical_therapy</Attribute>
<Attribute name="number_physical_therapy_sessions">8</Attribute>
<Attribute name="hospital_code">1</Attribute>
</Data>
<WorkflowModelElement>physical_therapy</WorkflowModelElement>
<EventType>complete</EventType>
<Timestamp>1998-01-19T00:00:00.000+01:00</Timestamp>
</AuditTrailEntry>
</ProcessInstance>
```

Figure N.2: MXML log fragment for the *therapies log*

O: Association rules with wide range of confidence values

Here we provide all the association rules generated by the Apriori algorithm for the event log used in Section 7.2.2, with a population size=50, confidence=0.5 for Illustration 7.2.2, Chapter 7. Nineteen association rules were obtained with confidence ranging from 0.95 to 0.73.

1. therapyAcutePhase_type20,therapyAcutePhase_type14=>physical_therapy (conf: 0.95)

2. therapyAcutePhase_type18=>medical_complication_13 (conf: 0.92)

3. physical_therapy=>therapyAcutePhase_type14 (conf: 0.91)

4. therapyAcutePhase_type1,medical_complication_13=>physical_therapy (conf: 0.9)

5. medical_complication_6=>therapyAcutePhase_type20 (conf: 0.89)

6. medical_complication_17=>therapyAcutePhase_type1, therapyAcutePhase_type20 (conf: 0.89)

7. therapyAcutePhase_type14=>therapyAcutePhase_type1 (conf: 0.88)

8. physical_therapy=>therapyAcutePhase_type1, therapyAcutePhase_type20 (conf: 0.86)

9. medical_complication_13=>therapyAcutePhase_type1, therapyAcutePhase_type20 (conf: 0.86)

10. medical_complication_17=>therapyAcutePhase_type20, medical_complication_13 (conf: 0.85)

11. therapyAcutePhase_type20=>therapyAcutePhase_type1 (conf: 0.83)

12. therapyAcutePhase_type13=>therapyAcutePhase_type1, therapyAcutePhase_type20 (conf: 0.81)

13. therapyAcutePhase_type15=>therapyAcutePhase_type1 (conf: 0.81)

14. therapyAcutePhase_type14=>therapyAcutePhase_type20 (conf: 0.77)

15. therapyAcutePhase_type1=>therapyAcutePhase_type20 (conf: 0.76)

16. therapyAcutePhase_type20,medical_complication_13=>physical_therapy (conf: 0.75)

17. therapyAcutePhase_type20=>therapyAcutePhase_type13 (conf: 0.74)

18. physical_therapy=>therapyAcutePhase_type20, medical_complication_13 (conf: 0.73)

19. therapyAcutePhase_type18=>therapyAcutePhase_type20 (conf: 0.73)

P: Technical architecture of the ARM

1. The purpose of the ARM

1.1 Introduction

This appendix provides an overview of the technical architecture for the Association Rule Miner (ARM) plug-in implemented in the Process Mining framework. The ARM plug-in generates frequent itemsets and association rules between the activities recorded in an event log. In this document we refer to the development effort for ARM as the 'project'.

1.2 Goals of the project

The implementation of the ARM is viewed as an attempt to gain insights into the flexible and less structured healthcare processes. The motivation behind this development effort can be read in detail in the Chapter 4, Section 4.2.

2. Stakeholders of the project

The stakeholders of the ARM project mainly includes the researchers, students and process analysts interested in mining less structured processes and gaining insights into these processes. These stakeholders must be well acquainted with the process mining concepts, characteristics of less structured processes and association rules.

3. The scope of the ARM

This section describes the use case of the system i.e. the ARM plug-in.

Actors:	Researchers/Students/Process Analysts
Description:	The user invokes the ARM mining plug-in and obtains frequent itemsets and/or association rules. This can be further used to cluster the process instances of the event log.
Trigger:	The user invokes the ProM framework and is interested in mining plug-ins that are meant to mine less structured/flexible processes.
Preconditions:	1. The user has ProM installed on his PC. 2. The user selects the *mining plug-ins* option in the ProM.
Normal Flow:	1. Convert the MXML log in the ARFF format accepted by Weka. 2. Select the association rule algorithms (the Apriori or the PredictiveApriori) 3. Set the parameter settings for the selected algorithm. 4. Select whether the frequent itemsets should be obtained. 5. Obtain association rules and/or frequent itemsets. 6. Use association rules and/or frequent itemsets for clustering the log.
Post conditions:	1. Obtain association rules and/or frequent itemsets. 2. Use clusters for further mining/analysis plug-ins in ProM.
Alternative Flows:	1. In parallel to Steps 2-4, the user can save the learning instances in the ARFF format to a desired location. The user can chose to quit the plug-in after this.

4. Use cases

In the Figure P.1, the use cases that were defined for the ARM plug-in are depicted. In these use cases only one general actor is shown i.e. user. This user can be anyone of the above mentioned stakeholders. We identify four main use cases for the ARM plug-in:

1. Convert MXML log to ARFF
2. Generate and retrieve frequent itemsets
3. Derive association rules

4. Cluster the PIs on the basis of FIS or Rules

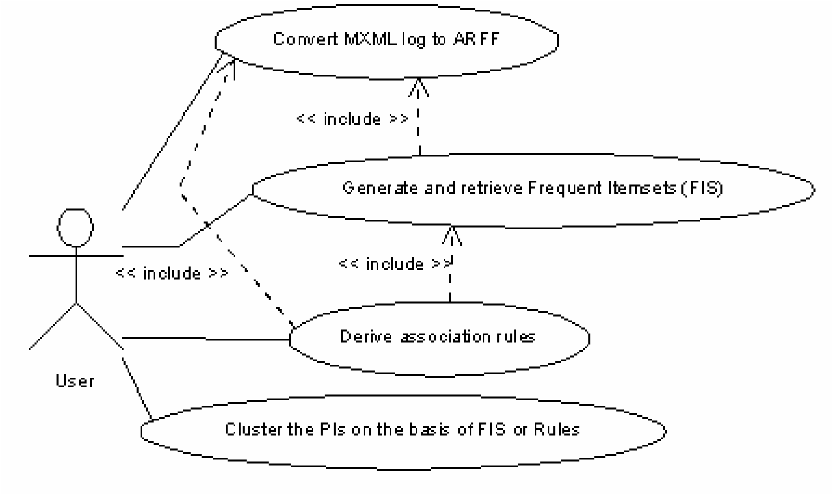


Figure P.1: Use cases *Association Rule Miner*

4.1 Convert MXML log to ARFF

Description:	The user invokes the ARM mining plug-in and before the mining starts, the input MXML log is converted to the ARFF file.
Trigger:	The user invokes the ProM framework and is interested in mining plug-ins that are meant to mine less structured/flexible processes.
Preconditions:	1. The user has ProM installed on his PC. 2. The user selects the *mining plug-ins* option in the ProM.
Normal Flow:	1. The user chooses the ARM mining plug-in and clicks the button *start mining*.
Post conditions:	1. The learning instances obtained in the ARFF format can be saved to the desired location. 2. The saved file can be used for experimenting with algorithms provided in the Weka machine learning library.
Alternative Flows:	-

4.2 Generate and retrieve frequent itemsets (FIS)

Description:	The user invokes the ARM mining plug-in and clicks the *start mining* button. The user is interested in retrieving the frequent itemsets (which are further used to generate association rules).
Trigger:	The user invokes the ProM framework and is interested in mining association rules that depict correlation between activities in the event log given as input to ProM.
Preconditions:	1. The user has ProM installed on his PC. 2. The user selects the Apriori algorithm from the ARM and clicks the *start mining* button.
Normal Flow:	1. The user chooses the Apriori algorithm for the association rules. 2. For retrieving the frequent itemsets, the user selects the option *output frequent itemsets?*
Post conditions:	1. The frequent itemsets are generated and the user retrieves them in an output window. 2. These itemsets can be used for clustering the event log to retrieve process instances containing the itemset.
Alternative Flows:	Step 2 can be omitted if the user does not want to retrieve the frequent itemsets.

4.3 Generate association rules

Description:	The user invokes the ARM mining plug-in and the plug-in generates association rules on the basis of the algorithm selected by the user.
Trigger:	The user invokes the ProM framework and is interested in mining association rules that depict correlation between activities in the event log given as input to ProM.
Preconditions:	1. The user has ProM installed on his PC. 2. The user selects the Apriori algorithm or the PredictiveApriori algorithm from the ARM and clicks the *start mining* button.
Normal Flow:	1. The user chooses the Apriori/PredictiveApriori algorithm for the association rules.
Post conditions:	1. The association rules are generated. 2. These rules can be used for clustering to retrieve the process instances satisfying the rule.
Alternative Flows:	-

4.4 Cluster the PIs on the basis of FIS or Rules

Description:	The user clusters the event log on the basis of a particular frequent itemset or association rule.
Trigger:	The user invokes the ProM framework and is interested in obtaining process instances specific to a rule or an itemset.
Preconditions:	1. The user has ProM installed on his PC. 2. The user obtains association rules from the Apriori algorithm or the PredictiveApriori algorithm and frequent itemsets from the Apriori algorithm.
Normal Flow:	1. The user selects an association rule or a frequent itemset and clicks the *cluster* button.
Post conditions:	1. Process instances are grouped into clusters satisfying the selected rule or itemset. 2. The user also receives the count of PIs satisfying the rule or itemset. 3. The clustered PIs can be used for further mining/analysis algorithms..
Alternative Flows:	-

5. User requirements

In this section, the requirements from the Association Rule Miner plug-in in ProM are listed along with their priority. Priorities have been assigned to requirements based on the relevance of the functionality. The following priority levels exist:

1. The requirements of this highest priority level must be met by the ARM. These requirements may be the requirements directly related to association rule algorithms, or the requirements that aid a user in using the plug-in easily.

2. The requirements of this priority level relate to the additional functionality that the plug-in offers besides generating frequent itemsets and association rules.

5.1 Functional requirements

Requirement ID	Requirement	Priority
FR1	The MXML log must be converted to the ARFF format.	1
FR2	The plug-in must generate rules/itemsets on the basis of user's parameter settings.	1
FR3	The user if interested must be able to view the frequent itemsets.	1
FR4	The user must be able to retain the non-redundant rules from the original rules obtained from the Apriori algorithm.	1
FR5	The user must be able to select any itemset or rule to cluster the event log.	2

FR6	The user must be able to export the selected PIs (in a cluster) to a new log file or he must be able to use the selected PIs for further mining/analysis.	2
FR7	The user must be able to retrieve the learning instances in the ARFF format obtained by converting the MXML log.	2
FR8	The user can retain the event type information in the generated itemsets and rules.	2

5.2 Non functional requirements

- **Look and Feel Requirements:**

Requirement ID	Requirement	Priority
NFR1	The Association Rule Miner must be added as a mining plug-in in the ProM framework.	1
NFR2	An MXML log must be given as input to the ARM.	1
NFR3	The user interface of ARM must be clearly visible and the colours used must be synchronized with those used by other plug-ins in ProM.	1

- **Usability requirements**

Requirement ID	Requirement	Priority
NFR4	The ARM must be easy to use and the potential user must be able to learn using it quickly.	1
NFR5	The ARM must be accompanied by user manuals integrated in the Plug-in Help System of the ProM framework.	1

- **Reliability & Availability requirements**

Requirement ID	Requirement	Priority
NFR6	Once the user has installed ProM on his PC, the ARM must be available and accessible 24x7.	1

- **Scalability requirements**

Requirement ID	Requirement	Priority
NFR7	The ARM must be scalable and the user with appropriate permissions and expertise must be able to extend the functionality offered by it.	1

- **Quality requirements**

Requirement ID	Requirement	Priority
NFR8	The source code of the ARM must conform to the coding standards used in the ProM research group.	1

6. Technical diagrams

In this section we give technical diagrams that showcase the static and dynamic behaviour of the ARM plug-in. We chose Unified Modelling Language (UML) diagrams for this purpose. UML *class diagrams* were used to represent the structure and substructure of the ARM through objects, attributes, operations and relationships. Class diagram is a structure diagram and it does not show the behaviour of the plug-in. Therefore, we chose *activity diagrams* to show the behaviour of the ARM.

6.1 Static view of the ARM

The static view of the plug-in is shown by the class diagram given in the Figure P.2. This class diagram shows the various classes representing the structure of the plug-in and, the relationships between these classes shows how these classes will interact with one another. In the diagram below, the dotted horizontal lines indicate the classes from the ProM framework, ARM plug-in and Weka library respectively.

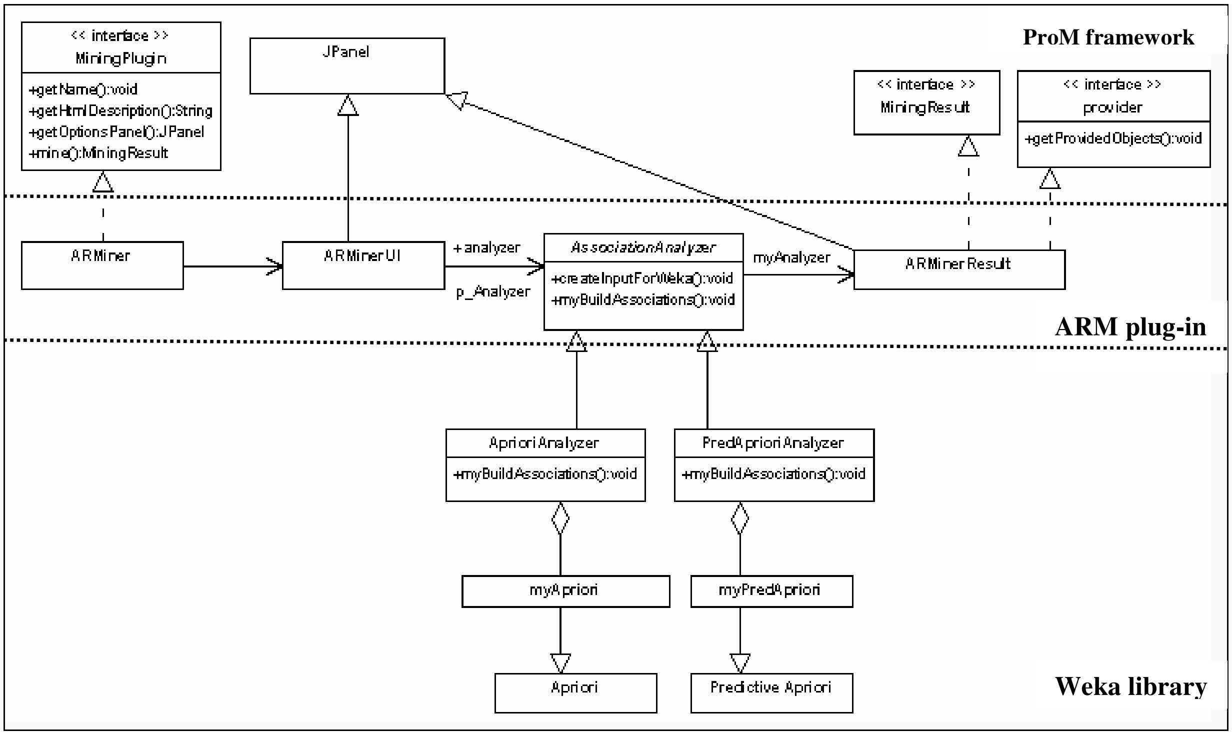


Figure P.2: Class diagram of the ARM plug-in implementation

All mining plug-ins in the ProM framework need to implement the MiningPlugin interface. As can be seen in Figure P.1, the ARMiner class implements the interface for the Association Rule Miner plug-in. The interface requires four methods:

- getName(): this method returns the name of the plug-in.
- getHtmlDescription: this method returns the user documentation in HTML.
- getOptionsPanel(): this method returns a JPanel containing the user interface for setting the options that are specific for this plug-in.
- mine(): this method is called as soon as the plug-in is invoked. It executes the mining algorithm.

When the mine method of the ARMiner is called, first the ARMinerUI, i.e. the GUI of the plug-in, is created.Through the GUI the user can select the association rule algorithm and set certain options (by giving certain values to parameter) before the association rule algorithm is executed. Before the user's selected algorithm is executed, the MXML log given as input to the ProM is converted into the ARFF format learning instances. This log is given as input to the CreateInputForWeka() method defined in the AssociationAnalyzer class. The format of converted MXML log can be seen in Appendix L. AssociationAnalyzer is an abstract class providing user-defined methods for the Apriori and the PredictiveApriori algorithms. Depending on the user's selection of the algorithm, the Apriori or the PredictiveApriori algorithm, myBuildAssociations() method of the AssociationAnalyzer is called.

AssociationAnalyzer class has a variable of type *Associator* which is an abstract scheme for all learning associations. If the user chooses the Apriori algorithm, then this *Associator* object in class AprioriAnalyzer is set as an object of MyApriori class. MyApriori class extends Apriori class, which is the class implementing the Apriori algorithm in Weka. Similarly, if the user chooses the PredictiveApriori algorithm, then this *Associator* object in class PredictiveAprioriAnalyzer is set as an object of MyPredApriori class. MyPredApriori class extends PredictiveApriori class, which is the class implementing the PredictiveApriori algorithm in Weka. myBuildAssociations() method calls the Weka method: buildAssociations(data) where data is the input in the ARFF format, supplied by the CreateInputForWeka() method. This method executes the actual association rule algorithm, and generates frequent itemsets and then association rules from them.

The association rules obtained from this method are stored in parts: antecedents and consequents. These LHS and RHS parts are used by the get_m_allTheRules() method defined in the MyApriori and MyPredApriori classes. We implement in this method our approach for retaining non-redundant rules (cf. Section 5.4) from the Apriori algorithm. Also, for both the algorithms this method contains the logic for displaying the rules in an appropriate format and not in the Weka format. The rules in Weka format can be seen, for example, in Figure 4.3 and 4.4 in Chapter 4. The class MyApriori also defines the method get_m_LS()containing the logic for proper display of frequent itemsets. For the PredictiveApriori algorithm, the frequent itemsets are dynamically created and destroyed (not stored in memory), hence this algorithm does not give the user the facility to retrieve these itemsets.

Further, as the user can also cluster the PIs on the basis of an association rule and a frequent itemset, this feature is implemented in classes MyApriori and MyPredApriori. The methods checkForFIS() and checkForRule() contain the logic for checking whether a PI satisfies a particular (user-selected) association rule or contains a particular (user-selected) frequent itemset. These classes also contain methods that take into account the user's choice for retaining event type information for association rules and frequent itemsets.

Thus, based on the algorithm selected and the user's input for different parameters the result: association rules and frequent itemsets, is returned to the framework as a GUI component i.e. ARMinerResult. ARMinerResult is derived from JPanel and is displayed in a separate window component. It also implements the Provider and MiningResult interface to export the selected log and to return the output as an instance of the MiningResult respectively.

6.2 Dynamic view of the ARM

After explaining the static view of the ARM through the class diagram in Figure P.2, we present the dynamic and behavioural view of the plug-in by an activity diagram given in Figure P.3. An activity diagram is an object-oriented equivalent of flowcharts and data-flow diagrams.

The activity diagram in Figure P.3 shows that a user after invoking the ARM can set the parameters of the desired algorithm: the Apriori or the PredictiveApriori. He can also save the learning instances in the ARFF format. At this point, he can simply chose to quit the plug-in after saving the ARFF file or he may opt for continuing with the actual mining done by the plug-in. Based on the parameters set for the Apriori algorithm , he can opt for retrieving the frequent itemsets or retrieving both the frequent itemsets and association rules. For the PredictiveApriori algorithm, the user can only obtain the association rules. After he obtains rules and/or itemsets he can use the clustering feature to obtain cluster of process instances. At this point, he can just quit the ARM or continue using the cluster obtained for other mining/analysis plug-ins. Finally the user closes ProM and quits.

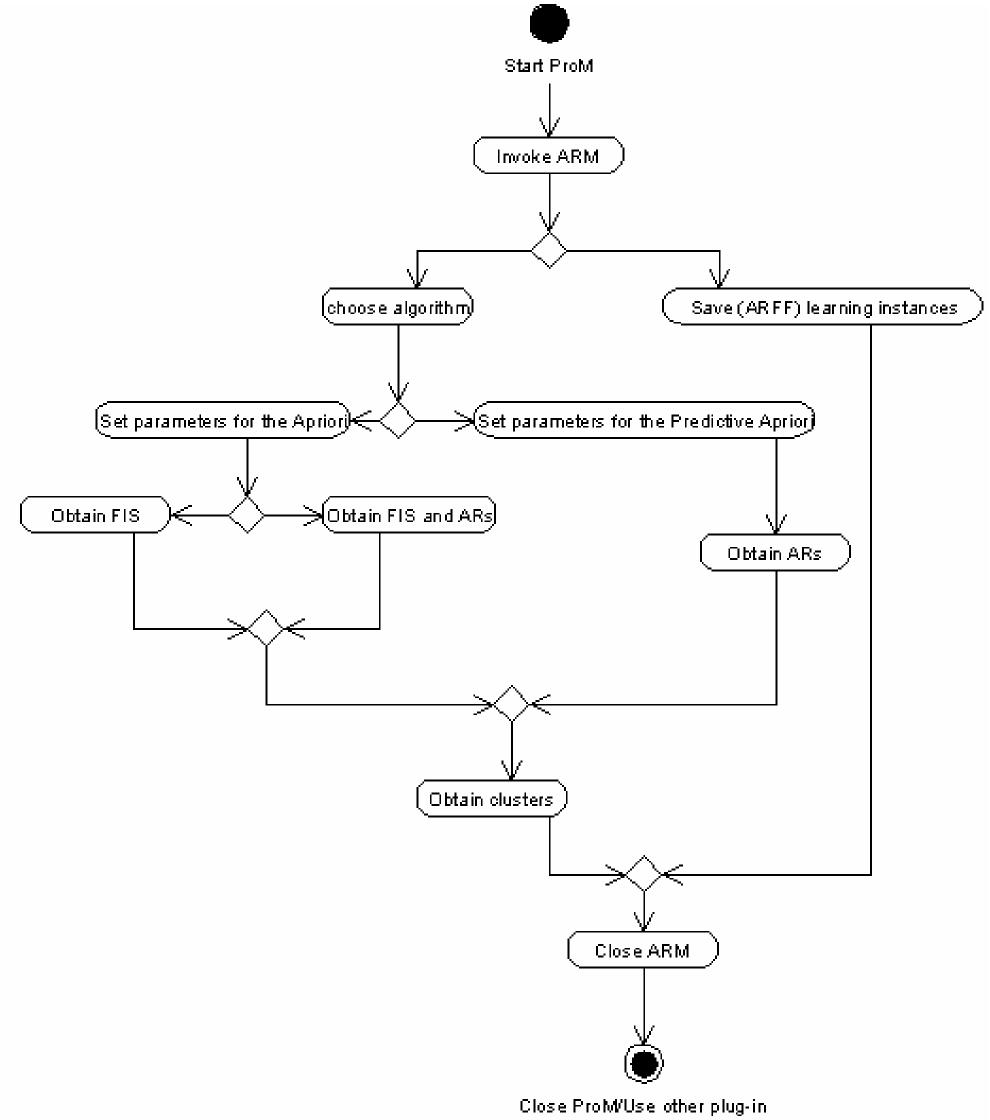

Start ProM
Invoke ARM
choose algorithm
Save (ARFF) learning instances
Set parameters for the Apriori
Set parameters for the Predictive Apriori
Obtain FIS
Obtain FIS and ARs
Obtain ARs
Obtain clusters
Close ARM
Close ProM/Use other plug-in

Figure P.3: Activity diagram of the ARM plug-in implementation