A VETERINARY DIAGNOSIS EXPERT SYSTEM FOR REMOTE USE

Submitted in partial fulfilment
of the requirements of the degree of

BACHELOR OF SCIENCE (HONOURS)

of Rhodes University

Stephen James Carse

_ Grahamstown, South Africa _
5th November 2013
Abstract

Expert system development has primarily been focused on applications designed to be run on personal computers. An expert system capable of diagnosing illness in livestock and accessible on low-end mobile phones in rural areas without strong cellular coverage could enable livestock owners living in such areas to access medical assistance for their animals which was previously inaccessible to them. An expert system capable of diagnosing illness in cattle was built upon the Drools rule engine, and a web-based interface was developed for it. This interface was designed to be useable on low-end mobile phones, given the constraints of screen resolution, processing capabilities, and the limited feature-set of their browsers, as well as in areas with poor cellular coverage. Under testing, the system was able to correctly diagnose a series of hypothetical test cases, and avoid diagnosing an illness under conditions in which it could not confidently do so. The interface of the system was tested on a variety of mobile browsers and was functional in each.
ACM Computing Classification System Classification

Thesis classification under the ACM Computing Classification System (2012 version, valid through 2013):

**H.3.4** [Decision support systems]: Expert systems

**J.4.3** [Ubiquitous and mobile devices]: Mobile devices

**L.4.5** [Health care information systems]

**General-Terms**: Human Factors, Verification
I would like to thank my supervisor, Karen Bradshaw, for her help and guidance during this project, as well as for all her support and understanding. I am indebted to her for helping to make this thesis what it is. A broader thank you to the Department of Computer Science’s staff for the education I have received over the past four years is also well-deserved.

Thank you to my parents for their love and support, as well as for affording me the opportunity to pursue my dreams at an institute such as Rhodes. Thanks also to my friends for their help and understanding this year.

I would like to acknowledge the technical support of Telkom, Tellabs, Stortech, Genband, Easttel, Bright Ideas 39 and THRIP through the Telkom Centre of Excellence in the Department of Computer Science at Rhodes University.


Contents

1 Introduction 1

1.1 Objectives of the Research 1

1.2 Problem Approach 2

1.2.1 Review of Literature 2

1.2.2 Design 3

1.2.3 Implementation and Testing 3

1.2.4 Interface Finalisation 3

1.3 Limitations of the Research 3

1.4 Thesis Organisation 3

2 Project Background 5

2.1 Aspects of the Project 5

2.1.1 Expert Systems 5

2.1.2 The Use of Mobile Phones 6

2.1.3 Rural Farmers 7

2.2 Cattle Ownership in the Rural Eastern Cape 9

2.2.1 The Importance of Cattle 9
2.2.2 The Economic Situation ........................................... 10
2.2.3 Mortality Amongst Rural Cattle ................................. 10

2.3 Expert System Shells ................................................. 10
   2.3.1 RETE Algorithm ............................................... 11
   2.3.2 Jess .............................................................. 12
   2.3.3 Drools .......................................................... 13
   2.3.4 Protégé .......................................................... 13
   2.3.5 Integrating Protégé with Jess ................................. 14

2.4 Knowledge Acquisition Process .................................... 14

2.5 Mobile Interface ..................................................... 17
   2.5.1 Website .......................................................... 18
   2.5.2 Native Application .............................................. 19
   2.5.3 SMS Based Application ....................................... 22

2.6 Summary .............................................................. 23

3 Design and Implementation of the CDS ............................. 24

3.1 Overview of the Expert System .................................... 24

3.2 Knowledge Acquisition .............................................. 25
   3.2.1 Choice of Knowledge Source .................................. 26
   3.2.2 Information in the InfoPaks .................................. 27
      3.2.2.1 Document Structure ...................................... 27
      3.2.2.2 Style of Language ....................................... 27
      3.2.2.3 Use of Illustrations ..................................... 28
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2.3</td>
<td>Information Extracted for use in the CDS</td>
<td>28</td>
</tr>
<tr>
<td>3.2.3.1</td>
<td>Symptom Information</td>
<td>29</td>
</tr>
<tr>
<td>3.2.3.2</td>
<td>Information on the Susceptibility of Animals to the Disease and Vaccination Procedures</td>
<td>30</td>
</tr>
<tr>
<td>3.2.3.3</td>
<td>Information about the Spread of the Disease</td>
<td>30</td>
</tr>
<tr>
<td>3.3</td>
<td>The Inference Engine</td>
<td>31</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Obtaining General Information</td>
<td>31</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Clarifying Broad Symptoms</td>
<td>31</td>
</tr>
<tr>
<td>3.3.2.1</td>
<td>Strategy and Considerations</td>
<td>31</td>
</tr>
<tr>
<td>3.3.2.2</td>
<td>Explanation of the Categories Used</td>
<td>34</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Determining Immunity</td>
<td>35</td>
</tr>
<tr>
<td>3.3.4</td>
<td>Interaction with Other Sick Animals</td>
<td>35</td>
</tr>
<tr>
<td>3.3.5</td>
<td>Vaccination History</td>
<td>36</td>
</tr>
<tr>
<td>3.3.6</td>
<td>Initial Possible Disease List</td>
<td>36</td>
</tr>
<tr>
<td>3.3.7</td>
<td>Rulefire</td>
<td>37</td>
</tr>
<tr>
<td>3.3.8</td>
<td>Querying Symptoms</td>
<td>37</td>
</tr>
<tr>
<td>3.3.9</td>
<td>Evaluation of Likelihood</td>
<td>37</td>
</tr>
<tr>
<td>3.3.10</td>
<td>Differentiation Process</td>
<td>37</td>
</tr>
<tr>
<td>3.3.11</td>
<td>Diagnosis</td>
<td>38</td>
</tr>
<tr>
<td>3.4</td>
<td>Conversion of Knowledge to Rules</td>
<td>38</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Diseases Chosen</td>
<td>38</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Initial Possible Diseases</td>
<td>38</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Querying of Symptoms</td>
<td>39</td>
</tr>
</tbody>
</table>
3.4.4 Evaluation Process .................................................. 42
3.4.5 Differentiation of Similar Diseases .............................. 44
3.4.6 Diagnosis ............................................................. 44
3.5 HTTP Servlet ........................................................... 46
3.6 Mobile Interface ....................................................... 48
   3.6.1 Choice of Interface .............................................. 48
   3.6.2 Basic Structure of the Interface ............................... 49
   3.6.3 Interaction using Dynamic Forms ............................ 49
   3.6.4 Index Page ....................................................... 50
   3.6.5 Initial Queries .................................................. 52
   3.6.6 Symptom Queries .............................................. 52
   3.6.7 Diagnosis ....................................................... 54
3.7 Logging Functionality ................................................. 54
   3.7.1 Logging During the Diagnostic Session .................... 56
   3.7.2 Viewing the Logs ................................................ 56
3.8 Implementation Specifics ............................................. 56
   3.8.1 Initial Phases and Setup ...................................... 56
      3.8.1.1 BroadSymptoms ......................................... 56
      3.8.1.2 Cow ..................................................... 57
      3.8.1.3 HerdInfection, Immunity and Vaccination ............ 57
   3.8.2 Symptoms and Diagnosis ...................................... 57
      3.8.2.1 Diagnosis ............................................... 57
      3.8.2.2 DiagnosisHelper ....................................... 57
3.8.2.3 Differentiator ........................................... 57
3.8.2.4 Possible Disease ....................................... 58
3.8.2.5 Suspected Diseases ................................. 58
3.8.3 Communication ........................................... 58
  3.8.3.1 Next .................................................. 58
  3.8.3.2 Queries ............................................... 58
3.8.4 Servlet Classes ........................................... 58
  3.8.4.1 CowVetServlet ........................................ 58
  3.8.4.2 HelperFunctions ..................................... 59
  3.8.4.3 KnowledgeSession ................................. 59
3.8.5 Rule Control .............................................. 59
  3.8.5.1 Processed ............................................ 59
  3.8.5.2 RuleGrouper .......................................... 59
3.8.6 Logging Functionality .................................. 60
  3.8.6.1 Logger ............................................... 60
  3.8.6.2 LoggerItem .......................................... 60
3.9 Summary .................................................... 60

4 Results and Analysis ......................................... 61
  4.1 Symptom Reference Table ................................ 61
  4.2 Results of Diagnostic Tests ............................. 61
    4.2.1 Basic Diagnosis Functionality .................. 63
    4.2.2 Vaccination does not Override an Overwhelming Presence of Symptoms .......................... 64
CONTENTS

4.2.3 Successful Differentiation: Asiatic Redwater and Heartwater . . . . 65
4.2.4 Unsuccessful Differentiation: Heartwater and Asiatic Redwater . . 66
4.2.5 Unsuccessful Differentiation: Gallsickness and Asiatic Redwater . . 66
4.2.6 Unsuccessful Diagnosis: Insufficiently High Confidence . . . . . . . 67
4.2.7 Age as a Factor in the Diagnosis . . . . . . . . . . . . . . . . . . . . . 68
4.2.8 Similar Symptoms for Non-Similar Diseases: No Diagnosis . . . . . 69
4.2.9 Similar Symptoms for Non-Similar Diseases: Diagnosis Made . . . . 70
4.2.10 The Effect of Immunity on the Diagnostic Process . . . . . . . . . 71
4.3 Evaluation of the UI . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 72
4.4 Analysis of the CDS . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 75
4.5 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 75

5 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 76
5.1 Revisiting the Objectives . . . . . . . . . . . . . . . . . . . . . . . . . . . . 76
5.2 Future Work on the Expansion of the CDS . . . . . . . . . . . . . . . . . 77

References . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 77
List of Figures

2.1 The basic structure of an expert system, should JESS be used in its development and assuming a remote-based implementation. Adapted from (Quesada & Jenkins, 2013). .................................................. 12

2.2 The structure of an expert system utilising Jess, Protégé and JessTab (Shue et al., 2009). ................................................................. 15

2.3 Pig-Vet in images only mode during a diagnostic consultation. The interface is designed as a web page to be viewed in a personal computer’s web browser (Zetian et al., 2005). ............................................... 19

2.4 An image highlighting the common characteristics across the web-browsers on different platforms, such as the back button being present, despite the vastly different user interfaces of the operating systems themselves (Charland & Leroux, 2011). .................................................. 20

2.5 Worldwide mobile operating system market-share at the end of March 2013 (Gartner, 2013). ......................................................... 20

2.6 Vodacom smartphone mobile operating system market-share at the end of March 2013 (MyBroadband, 2013). ................................. 21

2.7 An image of a proposed nutritional diagnosis expert system (Quesada & Jenkins, 2013), developed as a native Android application. ............... 22

2.8 An image demonstrating how, despite all three images being of the same application (WhatsApp), the interfaces are vastly different and conform to the user interface guidelines for their host operating systems (iOS, Windows Phone and Android, respectively) ............................................. 23
3.1 Overview of the structure of the expert system developed. The Google Chrome logo is used to signify web browsers and is not indicative of the use of Chrome as the client being a requirement to use the CDS. .......................... 25

3.2 Symptom information on Redwater, a tick-borne disease (Turton, 1999). .......................... 28

3.3 Illustration used to provide clarity on some of the symptoms for Lumpy Skin Disease (Thomas, 2013). ................................................................. 28

3.4 Illustration used to show the benefits of vaccinating cattle for Lumpy Skin Disease (Thomas, 2013). ................................................................. 29

3.5 A high-level view of the diagnostic process. ................................................................. 32

3.6 An example comparing a diagnosis made with the raising of confidence levels to one without this step performed. .................................................. 43

3.7 An example showing the value of the “>= 60” condition. ................................................ 43

3.8 The basic structure of the UI. ................................................................. 49

3.9 The index page which greets the user when they visit the system. .......................... 51

3.10 The broad symptoms query form. ................................................................. 53

3.11 An example of a form querying some symptoms. .................................................. 55

3.12 The response after a diagnosis has been made. .................................................. 55

4.1 The home page of the CDS requesting basic information about a cow on Opera Mini (left) and Nokia Xpress Browser (right). .................. 74

4.2 The vaccination query page of the CDS requesting basic information about a cow on Opera Mini (left) and Nokia Xpress Browser (right). .................. 74
List of Tables

3.1 Categories of Broad Symptoms used. .......................... 35

3.2 The rating system used to convert the calculated confidence score to a textual descriptor. ........................................ 45

4.1 Symptoms and the diseases to which they apply. .................. 62

4.2 Basic diagnosis functionality. ........................................ 63

4.3 Vaccination does not override an overwhelming presence of symptoms. .... 64

4.4 Successful: differentiation: Asiatic Redwater and Heartwater. .......... 65

4.5 Unsuccessful differentiation: Heartwater and Asiatic Redwater. .......... 66

4.6 Unsuccessful differentiation: Gallsickness and Asiatic Redwater. .......... 67

4.7 Unsuccessful diagnosis: Insufficiently high confidence. ................. 68

4.8 Age as a factor in the diagnosis. .................................... 69

4.9 Similar symptoms for non-similar diseases: no diagnosis. ............... 70

4.10 Similar symptoms for non-similar diseases: diagnosis made. ............. 71

4.11 The effect of immunity on the diagnostic process. ..................... 72
Chapter 1

Introduction

While large-scale farmers have access to veterinary professionals and expert advice, both for the general running of their farms and care of their animals and for the treatment of ill and injured animals, many developing farmers lack the required finances to consult these experts when the need arises. This situation is also true for many people living in rural communities and informal settlements who own a small number of livestock, and for whom these animals represent a significant portion of their wealth. In many instances when an animal belonging to someone in this position is ill, they may be unwilling to consult a vet owing to the cost involved, or may be entirely unable to do so, either as a result of a lack of finances or a lack of veterinarians in the area. An expert system accessible in remote areas capable of diagnosing ailments affecting animals or advising on the necessity of consulting a vet could greatly assist developing farmers and livestock owners. The system could potentially provide a front-line diagnosis and advise the animal’s owner on which further steps should be taken. Such a system could also assist in verifying the diagnosis of a vet or provide assistance to a vet in the diagnostic process. Owing to the prospective users of such a system not having access to a personal computer (PC) during the normal course of their working day or while attending to a sick animal, if at all, the system should be designed to be accessible through a simple interface on a mobile phone.

1.1 Objectives of the Research

To address the problem outlined above, we proposed the development of the Cattle Diagnostic System (CDS), accessible using a web-enabled mobile phone. The objectives of the research are as follows:
• Primary goals:
  
  – Develop an expert system for a specific veterinary domain
  – Provide an interface to interact with the expert system using a mobile phone
  – Provide a front-line medical diagnosis system for livestock to people who would otherwise be unable to access expert veterinary information, either at all or without difficulty or expense

• Secondary goals:
  
  – Provide a simple interface for an expert to add new information and knowledge
  – Automatically translate new information and knowledge to rules and add these to the knowledge base
  – Allow users and experts to provide feedback on previous diagnoses of the system
  – Take feedback into account in future diagnoses

1.2 Problem Approach

The system development was divided into four phases, namely literature review, design of the expert system, implementation and testing of the expert system and finalisation of the interface. These phases are outlined below.

1.2.1 Review of Literature

An exploration of the literature on the topic of expert systems, in particular those with a medical focus, was undertaken to gain an understanding of the field and what had been developed and produced thus far. The write-ups on previously developed expert systems were examined to glean information on both the development process and the acquisition of knowledge for the system. Various tools used to develop expert systems were also examined and evaluated.
1.2.2 Design

Following the review of literature a selection of a knowledge source for the expert information and the tools to be used to develop the system was made. These resources were used to test the feasibility of the decisions made regarding the structure of the expert system and the communication between the various components making up the system.

1.2.3 Implementation and Testing

Once the feasibility of the proposed system had been verified, the project shifted to the development of the system which was to meet the project objectives. Testing occurred throughout this process and choices made regarding the diagnostic approach were frequently reconsidered and adjustments made as a consequence of the results of the testing.

1.2.4 Interface Finalisation

Following the successful development of the portion of the system supporting the diagnosis process, the system interface was finalised and the text presented to the user was altered and completed to make it more user-friendly.

1.3 Limitations of the Research

The proposed research project is a proof of concept and, as such, the CDS was developed to provide the functionality required to demonstrate proof of the viability of the proposal outlined in this chapter, rather than be a fully-functional application able to be deployed for real-world usage. The field of application of the expert system was limited to a subset of animals and diseases, given the time constraints of the project.

1.4 Thesis Organisation

An outline of the structure of the remainder of this thesis is presented below:
Chapter 2 describes the literature reviewed in preparation for the development of the system. The chapter aims to give the reader an insight into the structure of expert systems and the methods of development, as well as provide a discussion on the mobile interface of the project.

Chapter 3 covers the design and implementation of the system developed to meet the stated project goals, termed the Cattle Diagnosis System.

Chapter 4 provides information on the testing and analysis of the functioning of the system.

Chapter 5 presents the conclusion of the thesis and a note on possible future work to extend the system.
Chapter 2

Project Background

This chapter provides a summary of the literature consulted in preparation for the development of the CDS. It covers a background explanation of expert systems, the use of mobile phones and the situations of rural farmers. An examination of the tools and methods for expert system development is then provided, followed by an overview of aspects of the process of acquiring the system’s expert knowledge. Lastly, an analysis of the South African mobile phone market is provided.

2.1 Aspects of the Project

2.1.1 Expert Systems

An expert system aims to record the knowledge of an expert in a particular domain and then emulate the thought processes of the expert in making decisions using the acquired knowledge in that particular field (MacDonald, 2010). It allows informed and reliable decisions to be made and conclusions to be reached on a matter in the absence of an expert to make the decisions (Giarratano & Riley, 2005). One of the key points of an expert system is that it is interactive (Chen et al., 2012), querying the user for information regarding the situation and then reasoning decisions based on this information, using its stored knowledge and heuristics. As it moves further toward a decision, more focused queries are addressed to the user regarding the situation.

The structure of an expert system consists of four primary components (Vogts, 2001), namely the knowledge base, working memory, inference engine and user interface (UI).
The knowledge base stores the knowledge possessed by the system, which is used to mimic the decision-making ability of the expert, and includes both firm facts and heuristics. The working memory contains information provided by the user during a particular consultation session and the conclusions reached thus far based on this information. The inference engine uses knowledge stored in the knowledge base and the information currently in the working memory to make new inferences and decisions. Lastly, the UI allows the user to interact with the system, both to allow the user to provide information about the problem and for the system to provide feedback to the user.

An expert system interacts with the user querying it, by requesting information about the problem being faced and making assumptions and inferences based on the information presented to it. As it nears a conclusion it requests more detailed and focused information to allow it to draw an ultimate conclusion (Giarratano & Riley, 2005).

The expert system also produces explanations of these solutions aimed at the user of the system (Bohanec & Rajkovic, 1990). It is important that this functionality is present, as studies have indicated that systems that provide an explanation of how a decision was arrived at are more likely to be implemented and regarded as useful (Bohanec & Rajkovic, 1990, McNee et al., 2006). In the same way that a human expert, such as a doctor, should be able to justify their conclusions when required to do so, so too should an expert be able to explain how it reached its conclusions. This allows the verification of the conclusions by an expert, where necessary (Negnevitsky, 2005, Giarratano & Riley, 2005).

### 2.1.2 The Use of Mobile Phones

In the past, expert systems that were developed were largely created for PCs, with network functionality not being considered, as this was often not present (Li et al., 2002). Since then, the popularity of the Internet has seen many expert systems shifting to becoming web-based and employing a server-client structure, if advantageous to do so, allowing for their remote updating and ensuring the latest information is utilised when a query is performed. The recent explosion in popularity of the mobile phone, and in particular the smartphone in the second half of the 2000s, has seen a rise in the number of expert systems with development targeted towards mobile phones as the expected devices for interaction with them.

A mobile-oriented expert system makes sense when considering the general working conditions of farmers. Sick animals are unlikely to be in the range of a desktop computer, or
2.1. ASPECTS OF THE PROJECT

within the easy reach and use of a laptop computer, making a system targeted at such
devices impractical. A system where the UI is presented on a mobile device and the pro-
cessing is sent to a remote server over the Internet would be more practical, as this would
provide the mobility required and also allow the system to be easily updated with the
latest knowledge and rules, as well as offloading the processing to a more powerful device
than a phone, many of which have limited processing abilities. Although the flagship
and high-end smartphones today do boast impressive processing power and storage\(^1\), it
is unlikely that the targeted users of the system would have ready access to such mobile
phones.

A downside to the offloading of processing and storage to a remote server is the resulting
necessity to transfer the results of the interaction between the user and the system over the
Internet. Considering the mobile devices anticipated to be used in accessing the system,
cognisance of the availability and quality of mobile Internet in the areas where the system
would generally be utilised is important. South Africa’s mobile operators claim to have
almost universal, with the exception of portions of the Northern Cape, coverage in the
country\(^2\), although this extends only to GPRS and EDGE. The availability advanced
technologies such as 3G and later iterations is indicated as being more sporadic in rural
areas of the Eastern Cape. In addition, operators tend to focus on urban areas when
upgrading infrastructure, because of the larger consumer demand for services in those
locations (Steyn, 2013). This indicates that a high-speed Internet connection will not
always be able to be relied upon, especially given the nature of mobile Internet, with
coverage and speeds differing greatly over relatively short distances in some location, as
can be observed through the general use of a mobile phone.

2.1.3 Rural Farmers

An analysis of the literature regarding expert systems previously developed can provide
insight into the potential challenges facing South Africa’s farmers and livestock owners. In
China, researchers developed a web-based disease diagnostic expert system for pigs (Pig-
Vet) (Zetian et al., 2005). In the beginning stages of the development, they interviewed
farmers in the rural region they were targeting (northern China) to get an idea of the
challenges farmers in the area faced. The overriding response from farmers was that


\(^2\)http://www.vodacom.co.za/personal/main/coverageMaps; http://www.cellc.co.za/network
coverage-map
existing systems, which had been developed for them to use, were overly intricate and they were often confused as to how to use the system. In a similar project, also in northern China, researchers asked fish farmers for their opinion regarding a fish disease diagnosis system (Fish Expert) which they were requested to evaluate (Li et al., 2002). The results also indicated that many tended to struggle to understand how to use the system and comprehend what the system was trying to ask them. A similar conclusion was reached in separate, unrelated studies after consultations with various other farmers, namely, wheat farmers in Pakistan (Khan et al., 2008), Jamaican coffee bean farmers (Mansingh et al., 2007), Australian rural business operators (Miah et al., 2009) and farmers located in the Taw catchment region of Devon in the United Kingdom (Oliver et al., 2012).

The researchers in the Pig-Vet and Fish Expert projects were able to draw up a list of important factors to consider when developing the systems, based on their interviews with farmers who would be the users once development had been finished (Li et al., 2002, Zetian et al., 2005):

- Farmers would prefer the system to have two broad goals, namely prevention and suppression. The former implies the containment of the outbreak of a disease and preventing more livestock from becoming infected, while suppression involves curtailing the effects of a disease on already infected animals as much as possible.

- The application should display different information to different users based on their level of experience in diagnosing ailments and their comfort in making diagnoses (MacDonald, 2010).

- The application should be intuitive and require little or no training for a person to become proficient in the use thereof.

- A multimedia rich interface is an advantage. Providing images of symptoms and infections assists many farmers, who indicated they experienced difficulty in identifying symptoms based purely on textual descriptions. A drawback with the multimedia interface cited was the time taken to load images in some instances where the Internet connection was poor, leading to frustration and hampering a speedy diagnosis (Li et al., 2002).

- Jargon should be avoided when interacting with inexperienced users. Should the user be a vet verifying a diagnosis, jargon should be used in place of simplified terminology.
2.2 CATTLE OWNERSHIP IN THE RURAL EASTERN CAPE

- Users should be able to provide feedback on their use of the system and the accuracy of diagnoses and decisions it reached.

- Once a disease has been identified, farmers would like to be provided with information on what the possible causes of the disease could have been and suggestions on how to avoid these in future, if possible.

- The relevant experts should be able to update the rule and knowledge base online, to allow for a current and well-maintained system, ensuring increasingly more accurate diagnoses as time passes and the system evolves.

- Farmers would like to be able to approach the problem from two angles: the knowledge that the animal is sick and ascertaining what ailment has afflicted it; and the knowledge that the animal is sick and suspecting a particular ailment and needing to verify that this is the case.

2.2 Cattle Ownership in the Rural Eastern Cape

Cattle were identified as the potential focus for the proposed system. This was as a result of the large number of cattle owned by people living in the poorer, rural parts of the country, particularly in the Eastern Cape, who could benefit from such a system. Over 40% of livestock units in less-developed areas of the country (Bembridge, 1986) are in the Eastern Cape and cattle owners on average own six head of cattle, which are kept largely for subsistence rather than commercial reasons. In rural areas of the Eastern Cape province, 76% of households use land for grazing purposes (Perret et al., 2000) and over 60% own cattle (Perret, 2002).

2.2.1 The Importance of Cattle

Cattle ownership is an important factor in the economic, social and spiritual spheres of life in areas such as the Transkei. In traditional African societies a man’s status and social security is directly tied to the number of cattle in his ownership; cattle are hence seen as an investment (Carver, 2007). More cattle from a small herd are slaughtered for ceremonial purposes, such as marriages and funerals, than are sold for financial gain (Bembridge, 1986, Carver, 2007). It was estimated in a 1986 study that ownership of ten head of cattle was required to meet the basic needs of survival in poor rural areas
2.3. EXPERT SYSTEM SHELLS

of the Eastern Cape, in addition to sociological factors such as lobola (the payment for a bride), weddings and funerals (Bembridge, 1986). The importance of cattle to the society indicates the likelihood of continued ownership of cattle among rural communities in future years. Although cattle are slaughtered in some instances for social occasions, many diseases which can cause the death of a cow render the meat unfit for human consumption or cause the animal to lose a lot of weight and damage the condition of its body, rendering the potential gain of food from the animal’s death nullified (Thomas, 2013, Department of Agriculture, 2012b,c).

2.2.2 The Economic Situation

The Eastern Cape is South Africa’s poorest province, with a large percentage, 27%, of households classified as impoverished and over 70% of its inhabitants classified as poor (Perret, 2002). Many rural non-commercial farmers are still affected by the legacy of apartheid-era discriminatory policies, which denied them access to financial and other resources. The average household income in the Transkei, a part of the Eastern Cape no longer officially termed thus but still colloquially referred to as such and which is widely regarded as one of the poorest parts of the country, in 2002 was R6000 per annum, with most residents indicating they were short of money and access to food (Perret, 2002). Government pensions and grants are relied on heavily in the region (Perret et al., 2000). The economic situation of many citizens in poorer communities indicates a financial inability to make use of a private veterinarian or other expert for assistance in the care and treatment, in the event of disease, of their livestock.

2.2.3 Mortality Amongst Rural Cattle

The same 1986 study discussed previously, observed a high mortality rate, far greater than the reproduction rate in many instances, of 17%, drastically higher than the rate among commercial farmers of 3% at the time (Bembridge, 1986).

2.3 Expert System Shells

A variety of tools, termed expert system shells, exist to facilitate the development of an expert system, to reduce the development time and to remove the need to develop

http://www.sowetanlive.co.za/news/2012/06/15/transkei-village-wins-war-against-poverty
the entire application from scratch (Owaied & Qasem, 2010). These tools consist of an inference engine and an interface to assist an expert in the construction of the knowledge base. The knowledge base and UI for the application are provided to varying degrees when a shell is used (Negnevitsky, 2005).

There are various factors to consider in deciding whether to make use of an expert system shell or to code the system from the ground up. The use of a shell provides the advantages of speeding up the development pace of the system and providing for simpler maintenance of the knowledge base (Saad, 2008). There are drawbacks, however, in the inference engine, as the reasoning cannot be customised, the UI, as some can be cumbersome and lack user-friendliness, and explanation facilities, as these cannot always be customised to specific requirements of the system being implemented. Programming the system manually does result in a longer and more complex development period, but means that all aspects of the system can be tailored to the particular system’s requirements (Saad, 2008). The use of a shell does not, however, mean that all the drawbacks mentioned will occur. Well-established shells can, in many cases, provide efficient performance and/or strong reporting functions while facilitating a drastically simpler development process. Many also allow for the development of a custom UI for the end-users of the application.

Before discussing two of the expert system shells widely used today, we give an explanation of the RETE algorithm used for constructing rule engines.

### 2.3.1 RETE Algorithm

The RETE algorithm is the de-facto algorithm for rule engines today and is used to find all possible matches when comparing a large set of patterns to a large set of objects (Forgy, 1982, Lee et al., 2000). The algorithm gains its widely accepted efficiency by not iterating over sets, instead employing a tree-structure for its operation (Forgy, 1982). If-then statements, on which rule-based systems are based, are checked to determine which of these can be executed, based on the current facts in the working memory. Efficiency is gained as the algorithm remembers, from each cycle of iteration over the rules to the next, which facts it has previously matched and only re-examines the rules affected by changes to the working memory caused by the addition, modification or removal of facts (Forgy, 1982, Lee et al., 2000). This mechanism results in the algorithm having a linear computational complexity. The matching process, by which facts are matched to rules, is performed by compiling the patterns into a program.
2.3. EXPERT SYSTEM SHELLS

Figure 2.1: The basic structure of an expert system, should JESS be used in its development and assuming a remote-based implementation. Adapted from (Quesada & Jenkins, 2013).

2.3.2 Jess

Jess (Java Expert System Shell) was developed by Sandia National Laboratories in the United States (Sandia National Laboratories, n.d.). Based on CLIPS (C Language Integrated Production System) and previously co-developed by NASA, Jess is a Java-based shell. Jess uses an improved version of the RETE algorithm - designed to be efficacious when comparing a large group of patterns to objects (Forgy, 1982) - to process rules, and as such is very efficient in pattern-matching (Giarratano & Riley, 2005, Jaques et al., 2013). The Jess developers have also extended the shell beyond the capabilities inherited from CLIPS. It possesses the ability to inference using forward and backward chaining and has enhanced maintenance efficiency (Shue et al., 2009, Eriksson, 2003). Jess provides strong integration with Java, and allows access to Java APIs, facilitating the direct manipulation and analysis of Java objects, as well as the creation of objects, calling of methods and implementation of interfaces without the compilation of any Java code (Sandia National Laboratories, n.d., Quesada & Jenkins, 2013).

Jess is provided on a proprietary license and the distribution of the source code is not permitted. An academic license is provided free of charge by the makers (Sandia National Laboratories, n.d.), but the company charges several thousand dollars for a commercial license (MacDonald, 2010). The use of Jess for the CDS would therefore have implications for the future viability of the system.
2.3.3 Drools

Drools, like Jess, is based on Java and is an open source business logic integration platform (Jaques et al., 2013, MacDonald, 2010, JBoss Community, n.d.). The platform is a JBoss community developed project and consists of four sub-projects. Drools Expert is one of these and is currently seen by some to be the premier open-source rule engine (Proctor, 2012). Drools Expert provides solid integration with Java, allowing easy integration with the language. The rule engine is stout and provides many options for the development of rules for the expert system (MacDonald, 2010).

Drools Expert uses its own modified version of the RETE algorithm, dubbed RETEOO, which is an object-oriented version of the algorithm. It matches the patterns of rules to the properties of objects and is persistent. This allows it to monitor the state of a collection of objects and behave appropriately (MacDonald, 2010). The rule engine employs a forward-chaining mechanism; facts are inserted into the working memory, rules are scheduled for execution based on these facts and the actions specified are taken. The Drools documentation states (JBoss Drools Team, n.d., p. 5): “Forward chaining is “data-driven” and thus reactionary, with facts being inserted into working memory, which results in one or more rules being concurrently true and scheduled for execution by the Agenda.” Drools Expert assigns a higher priority to the speed at which a result is obtained than memory usage and can consequently cause memory problems in very large systems (Wu et al., 2012).

Drools is open-source software (MacDonald, 2010), licensed under the Apache license, meaning that the use of the system for this project would not result in future complications with royalties or license agreements to continue using the software.

2.3.4 Protégé

An ontology is defined (Shue et al., 2009) in the knowledge engineering field as “a systematic analysis of knowledge of some domains of interest, so that it can be shared by others”. Another definition (Gruber, 1993) often cited is that “an ontology is a formal, explicit specification of a shared conceptualisation”.

The original goal of Protégé was to attempt to help reduce the bottleneck involved in the acquisition of knowledge for an expert system (Shue et al., 2009, Gennari et al., 2003). The thought process behind the first version of the software was designed around the idea
that the acquisition of knowledge should occur in steps and that, after a step, knowledge elicited during that step could be used to produce tools for knowledge-elicitation in ensuing stages (Gennari et al., 2003). It was intended to allow users to build other tools tailored to their specific needs and to help with the elicitation of knowledge for expert system knowledge bases. It has today evolved into a set of tools encompassing a wide range of functions.

Protégé is ontology based and is highly customisable, allowing users to develop their own structures and easily define connections between different items (Shue et al., 2009). It is extensible and allows for integration with other platforms, greatly increasing its appeal to developers of expert systems.

2.3.5 Integrating Protégé with Jess

Jess can be integrated with Protégé when developing an expert system (Shue et al., 2009). Protégé is used for this ontology-based structure of the expert system to construct the domain knowledge, while operational knowledge is constructed using the Jess rule system. As both Jess and Protégé are based on Java, they can communicate, through plug-ins, with other Java based systems. JessTab is a tool that has been developed to allow the two systems to communicate. Protege’s APIs can communicate with JessTab, enabling the inference engine in Jess to operate as though it was the inference engine of the combined expert system. In return, JessTab allows Jess to access Protege’s knowledge base when rules are fired. The structure of such a system is shown in Figure 2.2.

2.4 Knowledge Acquisition Process

The transfer of knowledge from an expert to the knowledge base is widely described to be the primary “bottleneck” (Owaied & Qasem, 2010, Gennari et al., 2003, Khan et al., 2008, Zetian et al., 2005, Li et al., 2002) in the development of any expert system. Attention is therefore often placed on this process in an attempt to aid the smooth transfer of the knowledge. It is important for the person translating the knowledge into rules to gain a good understanding of the problem domain, to ensure that trivial mistakes are not made and that they do not move in blindly when translating the rules into the syntax required by the knowledge base (Zetian et al., 2005).
Figure 2.2: The structure of an expert system utilising Jess, Protégé and JessTab (Shue et al., 2009).
In order to determine the needs of the farmers - the potential users of the completed system - questionnaires were distributed by the developers of the Pig-Vet system to 120 farmers to gather their opinions and needs (Zetian et al., 2005). Follow-up questionnaires were sent out to clarify any points of ambiguity or confusion, in particular where subjective answers could be provided.

The correct acquisition of knowledge is crucial to the eventual performance of an expert system. As such, the developers of the Pig-Vet system elected to follow a four-pronged approach to knowledge acquisition (Zetian et al., 2005):

- Experts in the swine veterinary field were interviewed in person. A large as possible group of experts was gathered to provide a varied pool of opinions as a form of self-validation of the knowledge elicited during these interviews. Any differences of opinion were resolved by a discussion within the group.

- Questionnaires were used to attempt to determine the quantity of subjective information used, as subjective information needs to be treated as such when developing the rules for the knowledge base. This type of questionnaire complements the interviews and helps to highlight any subjective discrepancies introduced in the knowledge acquisition process (Armoni, 1995).

- A web-based system was developed to allow approved experts and farmers to update the knowledge database. Upon an update being made, a knowledge engineer would verify the changes and modify the knowledge base as appropriate. Feedback was provided to the person who performed the change. This allowed the reasoning process to be validated by an expert and helped ensure that any modifications to the knowledge base did not introduce inaccuracies.

- The developers conducted a literature review on the diagnosis of swine diseases, providing an additional knowledge source while the system was being developed. The familiarisation of the knowledge engineer with the subject matter allowed for simpler, more effective discussions with the experts and farmers, as they had a working knowledge of the subject matter.

The developers of Fish Expert followed a similar approach (Li et al., 2002), similarly identifying the knowledge acquisition process as the bottleneck in the development process of any expert system. They stated that this is caused primarily by communication difficulties between the experts and knowledge engineers due to an understanding barrier between
2.5 MOBILE INTERFACE

the two. The knowledge engineer is often unable to effectively understand the complex knowledge the expert is trying to convey (Li et al., 2002). The Fish Expert developers also elected to interview and distribute surveys to the farmers targeted as users of the system to get an understanding of their needs, and also to obtain an idea of their experience in dealing with ailments afflicting their animals. Human experts were interviewed to acquire the expert knowledge for the system and a web-based knowledge elicitation system was similarly developed to allow the knowledge engineers and authorised experts to aggregate facts and translate these into rules for the knowledge base.

Various other expert system development reviews were studied and most (Chen et al., 2012, Mansingh et al., 2007, Khan et al., 2008, Miah et al., 2009) followed a similar theme to that discussed above, that is, interviewing experts - through personal interviews and questionnaires and surveys, and also interviewing the potential users of the system, both to elicit their thoughts and preferences and to obtain any knowledge they possessed in dealing with their respective charges. Literature reviews and web-based systems for the addition of further knowledge were also frequently mentioned. An additional point mentioned by one group of developers (Khan et al., 2008) is that walking experts through case studies so that knowledge engineers can examine the expert’s thought processes is beneficial to the knowledge elicitation process.

One paper (Oliver et al., 2012) sought to highlight the importance of not discounting the knowledge possessed by the potential users of the system and to involve them in the knowledge elicitation process. This could be done through methods such as interviews and questionnaires. The authors argued that although the farmers may not, for example, be qualified vets, a farmer working with their animals for a long period of time would have picked up a hands-on trove of knowledge regarding the care of these animals. They discovered the importance of visiting the farms themselves and of observing the work occurring there to verify the information collected through interviews and questionnaires. A farmer had informed them how they separated clean and dirty water for the watering of portions of the farm in a previous interview, but they observed a farm-worker emptying dirty water into a supposedly clean canal above a watering point. They emphasised the importance of verification of claims made as a result of this (Oliver et al., 2012).

2.5 Mobile Interface

As mentioned in Section 2.3, the UI forms a key part of the application. While the intricate development that is vital to the correct execution of the expert system takes place in the
rule base and inference engine, the UI will play an important role in determining the uptake of the system among its intended users, as this is the only component they will ever interact with. Careful consideration therefore needs to be given to the interface in selecting the most appropriate option.

There are three broad options available for interface development and these are elaborated upon below.

2.5.1 Website

A mobile website could be developed, allowing the expert system to be accessed through a web browser on most mobile smartphones. The development of a mobile website provides the following advantages:

- A mobile website is generally universally accessible across competing mobile platforms. While there are different versions of browsers on different operating systems, the development for these different browsers can be compared to developing for competing desktop browsers (Charland & Leroux, 2011). While there can sometimes be difficulties, these are generally able to be dealt with, unlike the incompatibility between native applications developed for different platforms. Here, just as an OS X application cannot run on Windows, an application developed for one platform cannot run on another.

- There are a limited number of differences between platforms that a developer is required to support (Charland & Leroux, 2011).

- A mobile website is available immediately when the user follows a link to the site, whereas a native application typically requires the user to install the application through their device’s app store (Summerfield, n.d.).

- Changes to the UI can be immediately pushed to users without the delay of an application approval process by the relevant store (Summerfield, n.d.). This advantage is negated in the case of Android, as Google Play does not require approval before updates are published. The user is, however, still required to download the update manually.

- A mobile website generally does not require specialised or high-end hardware to run. Most of today’s low-end feature phones\(^4\) are equipped with a colour screen

and mobile browser capable of browsing the Internet, provided basic HTML is used. Opera Mini is an example of a mobile browser aimed at features phones which has seen great success in allowing people without access to computers to browse the Internet and, midway through 2012, remained the world’s most popular mobile browser, despite the vast majority of its install-base consisting of low-end feature phones⁵.

### 2.5.2 Native Application

The primary advantage of developing a native application for the proposed system would be the fluidity and feel of the resultant UI, as the processing would still be done remotely and require a data connection. Whereas a website can feel disjoint from the operating system and be hampered by the unnecessary chrome of the browser’s interface, a native application can take advantage of the platform’s official Software Development Kit’s (SDK) native UI tools and, provided it conforms to the platform developer’s UI guidelines, feel part of the operating system itself (Charland & Leroux, 2011). This results in a familiar feel to the user, which can make the application feel more intuitive to use (Summerfield, n.d.). Native applications execute faster than web-based applications, although this advantage is normally more pronounced in processing intensive applications such as games (Charland & Leroux, 2011).

---

2.5. MOBILE INTERFACE

Figure 2.4: An image highlighting the common characteristics across the web-browsers on different platforms, such as the back button being present, despite the vastly different user interfaces of the operating systems themselves (Charland & Leroux, 2011).

Figure 2.5: Worldwide mobile operating system market-share at the end of March 2013 (Gartner, 2013).

The large drawback of developing a native application is that the system is concretely restricted to the platform it has been developed for (Charland & Leroux, 2011).

A potential benefit of a native application is local storage. This would allow the option for the multimedia aspects of the expert system to be downloaded to the user’s device while connected to a fast Internet connection, such as Wi-Fi, and then be called up natively from the application, falling back to downloading them if they have not yet been downloaded or if an update is pending.

Figures 2.5 and 2.6 demonstrate the worldwide and South African market share of smartphone platforms, respectively. Vodacom’s breakdown has been used to reflect the market share in South Africa in the absence of official data for South Africa. As Vodacom claims
2.5. MOBILE INTERFACE

Figure 2.6: Vodacom smartphone mobile operating system market-share at the end of March 2013 (MyBroadband, 2013).

approximately 53% of the South Africa cellular market (Tech Central, 2012), their data can reasonably be used as an extrapolation point from which to draw conclusions about the South African market as a whole across the different operators.

Bada and Symbian’s negligible presence can be discounted, leaving the four platforms depicted in Figure 2.6 as the competing candidates. Although BlackBerry claims an overwhelming portion of the South African market, the following reasons demonstrate that the development of an application native to their platform would be ill-advised:

- BlackBerry has released its new BB10 platform this year and it became available in South Africa at the end of February. Of the 3.1 million BlackBerry smartphones active on Vodacom’s network, the vast majority are legacy BBOS 5, 6 and 7. These platforms have been superseded by BB10 and the number of handsets running them and actively being used is expected to decline as older models are phased out and replaced by devices running the new OS.

- The flat rated Internet access is regarded as a primary reason for the high level of adoption of BlackBerry smartphones in South Africa, especially when compared to many foreign markets (Wilson, 2013). As BB10 does not make use of this service, this competitive advantage for the company has been removed for its future generations of devices.

- The current BB10 market in South Africa is more comparable to Windows Phone than to Android and it cannot be guaranteed that BB10 will reach nearly the same levels of dominance in the South African market as its predecessor systems.
2.5. MOBILE INTERFACE

Figure 2.7: An image of a proposed nutritional diagnosis expert system (Quesada & Jenkins, 2013), developed as a native Android application.

A major drawback for the advocacy of the development of a native application was the anticipated target audience of the system, who would be unlikely to own smartphones, despite the growth in smartphone deployment in recent years. While smartphone operating system developers, particularly Google and Microsoft, for their Android and Windows Phone platforms, respectively, have stated a desire to use cheap smartphones to allow the vast global population who have yet to use the Internet to do so\(^6\), entry-level smartphones are still priced higher than most feature phones.

Apple’s iOS was disregarded as a candidate given that the cost of their cellular offerings is out of range for the target audience of the application. The cost of a Mac computer, required for developing for iOS, is also prohibitively expensive.

2.5.3 SMS Based Application

The application could be made to be SMS based, with communication between the user (client) and the server occurring via SMS messages (Guthery & Cronin, 2002). As SMS operates over standard GSM, this would remove the potential drawback of a mobile website or application - as these rely on a data connection - and could pose a potential problem in areas with only EDGE coverage (Guthery & Cronin, 2002).

The use of an SMS based model would, however, drive the cost of using the service up, as both the client and server could potentially have to send multiple SMSs during the consultation, which would be more costly than data, particularly if data bundles are in place\(^7\). This type of system would also not allow for multimedia capabilities, as


\(^7\)http://www.cellc.co.za/cell-phone-prepaid/99c-for-real; http://www.cellc.co.za/smartdata-bundles
2.6 Summary

In this chapter, we have highlighted how a mobile based expert system could benefit farmers in remote, rural areas while their potential views and desires for the application have been shown. The prevalence of cattle ownership in rural South African communities, particularly in the former Transkei area has been considered. Expert system shells have been discussed and possible implementation options explored. Knowledge acquisition has been shown to be a critical aspect of expert system development and techniques have been explored to facilitate the process. Lastly, potential UI implementations have been briefly discussed.
Chapter 3

Design and Implementation of the CDS

The design and implementation of the system developed to achieve the project goals set out in Section 1.1, termed the Cattle Diagnosis System (CDS), is explained in this chapter. An overview of the CDS and its components is first presented, followed by information about the acquisition of knowledge for the expert system. An outline of the diagnostic process followed by the CDS is then provided followed by a discussion about the conversion of the acquired knowledge into rules. An explanation of the Servlet and Interface components of the CDS is provided, along with some information regarding the logging functionality of the CDS. Finally, some details specific to the implementation of the CDS are provided.

3.1 Overview of the Expert System

The system that has been developed to fulfil the goals outlined in Section 1.1 consists of several interacting parts, which combine to perform the functions of the expert system.

As shown in Figure 3.1, the majority of the components reside on the server-side, with only the UI located on the client-side. The inference engine and working memory components are provided by Drools Expert\(^1\), while the knowledge base is built at runtime by Drools from a set of Drools Rule Language (DRL) files, which were manually developed using

\(^1\)Despite the identifying of the Jess rule engine as the preferred rule engine, the delay between requesting and obtaining an academic license for its use to develop the CDS meant that Drools, also a robust and capable rule engine, was employed instead. More information about Drools can be found at http://www.jboss.org/drools/drools-expert.html
3.2. KNOWLEDGE ACQUISITION

The expert knowledge obtained and converted into rules. The UI consists of a JavaServer Page (JSP), which contains HTML and JavaScript code and is updated dynamically from the server throughout the diagnostic session. JQueryMobile\(^2\) is used to provide additional functionality to the interface.

Communication between the UI and the other expert system components is facilitated by a custom-developed servlet, which resides on an instance of JBoss Application Server\(^3\). This servlet receives HTTP requests from the interface, processes these, and places them into working memory. An update for the interface is obtained from the working memory, which is then sent via an HTTP response to the mobile client’s browser and displayed.

3.2 Knowledge Acquisition

The expert knowledge was obtained from publications provided by the South African government’s Department of Agriculture, Forestry and Fisheries\(^4\). These publications,

---

\(^2\)http://jquerymobile.com/

\(^3\)http://www.jboss.org/jbossas

3.2. KNOWLEDGE ACQUISITION

referred to by the Department as InfoPaks, are short documents compiled to assist
developing farmers in caring for their livestock and crops and contain basic information
on various diseases that could affect these livestock and the vaccination and proper care
for the treatment of these diseases. The documents have been compiled by the Animal
Health for Developing Farmers group and are aimed at small-scale farmers and livestock
owners without large capital expenditure who do not necessarily have access to veterinary
specialists. This is borne out by the use of illustrations, which are sometimes cartoon-like,
to accompany the text, rather than high-quality photographs, and references to obtaining
assistance from the state veterinarian in cases where a veterinary expert is required
(Department of Agriculture, 2012c, Turton, 1999). The documents are presented in basic
language without medical jargon and provide, in the case of those related to illnesses that
can afflict cattle, basic information about the symptoms which could manifest themselves
and the suggested follow-up action and treatment. Simple information regarding vaccina-
tion and precautionary steps to avoid the disease is also provided. The use of the InfoPaks
as the source of the expert knowledge for the system thus fulfills the goal of expanding
knowledge dissemination and providing an easy to use initial diagnostic system providing
a basic recourse to owners of livestock concerned about the health of an animal, without
seeking to replace veterinary experts themselves.

3.2.1 Choice of Knowledge Source

The InfoPaks were chosen as the knowledge source for the CDS as they allowed it to
be developed without encroaching on the professional territory of veterinary experts by
packaging information made available by the government to developing farmers to assist
them in an easier to use format. The placing of the information in the InfoPaks into an
expert system allows the user to query the CDS for symptoms and receive a diagnosis,
rather than having to hunt through the various InfoPaks, which are presented in a list
along with many other documents for other livestock and crops, for applicable diseases
and evaluate each document, possibly comparing several, before being able to reach a
conclusion. The documents are provided in PDF format, which can also present a problem
as these cannot always be easily viewed on feature phones and other mobile devices with
small size and low-resolution screens, whereas the CDS was designed to be able to scale
to this use-case. Some InfoPak documents exist in official South African languages other
than English, opening the door to the possibility of future extensions to the CDS to offer
multi-lingual support.
3.2. KNOWLEDGE ACQUISITION

3.2.2 Information in the InfoPaks

3.2.2.1 Document Structure

Documents about a disease contain similar basic information about the disease. The basic structure is outlined below, although each document does differ and some provide less information than others.

- Basic information about the disease and how it can affect livestock.
- Which animals can be affected, with information regarding increased or reduced susceptibility of cattle to the disease, based on factors such as age, and the time frame and conditions under which a disease is more likely to occur.
- The means by which an animal could be infected.
- Symptoms that could potentially manifest themselves.
- How the disease could be diagnosed. Many diseases can be initially diagnosed by examining the symptoms appearing in the animal, but require blood or other biological samples, such as skin, to provide a definite diagnosis.
- Information about which diseases could be confused with the disease because of similar symptoms.
- Treatment measures that should be taken once the disease has been identified and diagnosed.
- Information on vaccination against the disease and precautionary measures that could be applied to reduce the risk of the disease affecting livestock in future.

3.2.2.2 Style of Language

The language used to provide information in the documents is basic and lacks jargon. Where a term is used which could potentially not be understood by the intended reader, a brief explanatory sentence is provided. Figure 3.2 shows an example of the typical textual presentation of symptom information.
3.2. KNOWLEDGE ACQUISITION

Sick animals

- There are two types of redwater, namely Asiatric redwater and African redwater
- African redwater is the more serious disease, although African redwater is probably more important as it is more widely spread. The signs are the same, although nervous signs also occur with Asiatric redwater
- Fever (40.1 to 51.5 degrees C)
- Pale to yellow eyes and gums
- Rear cauterised urine
- With Asiatric redwater there could be nervous signs, with difficulty in walking and convulsions (can look like heatwater)
- Death
- Calves are resistant for the first 5 to 9 months of life

Figure 3.2: Symptom information on Redwater, a tick-borne disease (Turton, 1999).

Figure 3.3: Illustration used to provide clarity on some of the symptoms for Lumpy Skin Disease (Thomas, 2013).

3.2.2.3 Use of Illustrations

Low-detail outlined images are included throughout the documents to illustrate the symptoms and effects of the diseases. High quality photographs which could be used to compare described symptoms with those present in the ill animal are not provided. Figures 3.3 and 3.4 provide examples of the types of illustrations used.

3.2.3 Information Extracted for use in the CDS

Each document was examined to determine the information that could provide useful knowledge for incorporation into the expert system. The information typically included symptoms, vaccinations and information relating to factors that could increase or decrease the probability of a disease occurring, such as in which season the animal had been born or what the current season is. Many diseases spread by insects are more likely to occur after wet weather, when the birth-rate of the insects increases.
To provide an example, the following subsections describe the information extracted for use in the CDS for Lumpy Skin Disease (LSD).

3.2.3.1 Symptom Information

The following symptoms are described for LSD (Thomas, 2013):

- Skin nodules and ulcers
  - Can vary from a few to hundreds
  - Size ranges from 0.5 - 5 cm
  - Can occur anywhere on the skin, including the nose, in the mouth and vulva or scrotum

- Legs that become swollen and develop sores

- Enlarged lymph nodes

- Pneumonia/coughing - as a result of infection of the respiratory tract (the windpipe) and lungs

- Nasal discharge - thick, watery to pussy fluid from the nose

- Infertile bulls - due to orchitis (infection of the testes)

- Infertile cows
• Mastitis - this lowers milk production
• Lachrymation, infection of the eye or even blindness
• Fever
• Emaciation
• Salivation

3.2.3.2 Information on the Susceptibility of Animals to the Disease and Vaccination Procedures

The following information is provided to clarify which animals face varied risk of becoming infected (Thomas, 2013):

• Cattle of all ages can be infected
• Cattle that are vaccinated annually are protected and therefore less likely to be infected.
• Cattle that have had the disease and survived cannot be infected again (immune).
• Calves under six months of age are protected against the disease if their mothers were vaccinated or had the disease previously.

In addition to the above points, the document (Thomas, 2013) states that animals should be vaccinated annually to benefit from the increased protection the vaccination provides. A calf under six months of age is protected provided its mother was correctly vaccinated or has previously survived the disease. Calves must be vaccinated annually once they reach six months of age.

3.2.3.3 Information about the Spread of the Disease

The document (Thomas, 2013) states that biting flies, which occur most frequently during the wet season, generally summer and autumn, are mostly responsible for the transmission of LSD. Calves that drink milk from an infected cow can also be infected as well as cattle of all ages through infected saliva, from drinking from the same drinking trough.
3.3 The Inference Engine

The overall structure of the process the system follows when performing a diagnosis is shown in Figure 3.5. As shown in this figure, the diagnostic process is broken up into several phases which are discussed in detail below.

3.3.1 Obtaining General Information

The system requests the following information about the cow from the user:

- Age (in years and months)
- The sex of the cow
- Whether the cow produces milk
- Whether the cow is regularly vaccinated
- Whether the cow may have come into contact with (directly or indirectly) any other cow that has recently been diagnosed with a disease
- Whether the cow has previously been diagnosed with a serious disease and survived it

3.3.2 Clarifying Broad Symptoms

The categories of broad symptoms identified and explained below are presented to the user, who is requested to select which of those is present. The user is also prompted for information on whether there has been at least moderate rain in the past two weeks.

3.3.2.1 Strategy and Considerations

The diagnostic strategy decided upon was to first query the user for the existence of broad categories of symptoms, into which the actual symptoms of diseases themselves could be classified. This allows the system to identify possible diseases based on these broad symptoms and then query for details regarding the specific symptoms of these diseases
Figure 3.5: A high-level view of the diagnostic process.
later in the diagnostic process. The strategy was implemented to avoid the number of steps of questions the system would require the user to run through before a diagnosis could be reached.

A common approach in medical expert systems is to allow the matching of specific symptoms to cause the system to hone in on a particular disease. This approach was not followed for the CDS because of the limitations of the information worked with and the choice of interface. The InfoPaks do not specify the relative importance of various symptoms with regards to specific diseases. While the conclusion could reasonably be drawn that a key symptom in Redwater is red-coloured urine, this is not explicitly stated, nor is it stated which symptoms are key indicators of Heartwater, the disease to which it is most similar (Turton, 1999). The InfoPaks do not refer to each other, and hence it is not possible to establish from them the order of priority of symptoms within a disease or among diseases. The system therefore assigns an equal importance to all symptoms. This has an implication for the diagnostic process since the InfoPaks state that an animal is not required to show all the symptoms of a disease for it to be afflicted by that disease; it can show only one symptom of a disease but still be infected with it. This is especially true if the disease is still in its early stages, which in the case of some diseases is the only time the animal can be cured. It was therefore felt that a disease could not be excluded from contention for diagnosis purely because only a few of symptoms were present.

The implication of the CDS being unable to determine which disease to focus on based on key symptoms appearing was that should it not utilise the strategy of querying broad symptoms to determine a list of possible diseases that should be queried, each symptom for each disease would need to be queried before a disease could be ruled out. While vaccination, immunity and interactions with other sick animals could assist this process, the recent vaccination for a particular disease does not by default exclude the possibility of a different disease, for example, rendering these considerations by themselves unable to avoid a protracted diagnostic process.

The second consideration was that of the interface. As the CDS is aimed at developing farmers and people who, in general, could reasonably be expected not to have high-end mobile phones or to live and farm in areas with limited cellular coverage, in particular with regards to high-speed cellular Internet, one of the aims of the mobile interface was to be useful but compact, and not require large chunks of data to be sent and received over the network. The querying of one symptom at a time could potentially lead to delays in the eventual diagnosis, as many transmissions over the cellular network would be required. While these transmissions would be small, the continual submission of forms
and the waiting for the receiving of data from the expert system over the network could potentially become tedious. The addition of a few fields to the form for each transmission and the resultant reduction in the number of transmissions required could reduce this problem. The knowledge session would also need to be paused and resumed less often. A balance therefore needed to be found between the number and size of transmissions between the client and server sides.

The querying of all symptoms at once would relieve the slow stop-start feeling required if all symptoms were required to be queried. This would however not make for a well-developed expert system, as it would largely be devoid of an initial diagnostic strategy, and would become unwieldy with the addition of additional diseases, and hence additional symptoms.

The broad symptom strategy was therefore used. Should the system be expanded and a more precise strategy be required, a combination of broad symptoms and the querying of specific symptoms could be enforced, with the querying of most likely diseases first and the querying of similar diseases together, allowing for the system to halt once a threshold of matched symptoms for a particular disease had been reached and a diagnosis and differentiation, where required, be made before all diseases identified for investigation had been queried. The implementation of such a strategy would require more detailed knowledge sources, as symptom importance and other potential likelihood factor strategies would need to be known. This strategy was hence contrary to the goals of this particular problem solution, as one of the aims was to enable the easier dissemination of existing information, rather than to encroach on the working territory of other professionals.

3.3.2.2 Explanation of the Categories Used

The categories of broad symptoms used are shown in Table 3.1. At present, a disease requires only one broad symptom for which it has a possible symptom to occur to be marked for further evaluation. Were the system to be expanded, a diagnostic approach involving rounds of querying, with more likely symptoms being queried before others and a check to see if a diagnosis can be made after each round, could be applied. This could be used in conjunction with the alternate strategy outlined in the previous section or done independently. The categories of broad symptoms could also be adjusted accordingly.
3.3. THE INFERENCE ENGINE

3.3.3 Determining Immunity

If the user confirmed when inputting the general information that the cow had previously been diagnosed with a serious disease and survived it, a list of diseases the system can diagnose is presented and the user is requested to select if the cow has survived any of these. Should they be immune to a particular disease, that disease is excluded from those that could be under consideration in that particular diagnostic session.

3.3.4 Interaction with Other Sick Animals

Some diseases can be transmitted from an infected animal to a healthy animal, infecting the latter, such as in the case of LSD where the spread of the disease occurs through infected saliva. Other diseases, such as those borne by ticks, are not spread between animals, but the infection of one animal through a tick-borne disease indicates a greater possibility that another animal could be infected with the disease owing to the prevalence of ticks. Similarly an examination of the information documents for Bovine Tuberculosis (Department of Agriculture, 2012b), Botulism (Department of Agriculture, 2012a) and

<table>
<thead>
<tr>
<th>Category</th>
<th>Explanation</th>
<th>Examples of Symptoms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Behavioural changes</td>
<td>Unusual behaviour which is not normally observed or which is clearly indicative of a problem with the cow</td>
<td>Depression, nervous walking (not related to physical problem such as a broken leg), convulsions</td>
</tr>
<tr>
<td>Digestive and urinary problems</td>
<td>Problems with the digestive and urinary processes of the cow</td>
<td>Constipation, discoloured urine</td>
</tr>
<tr>
<td>Drop in milk production</td>
<td>Milk production is lower than usual levels</td>
<td>Lowered milk production</td>
</tr>
<tr>
<td>External physical signs</td>
<td>Symptoms which manifest themselves on the exterior of the cow, such as in sores</td>
<td>Skin lesions, discoloured gums, sores on the skin</td>
</tr>
<tr>
<td>Fever and coughing</td>
<td>A fever or coughing of various descriptions</td>
<td>Fever, frequent coughing, rasping coughing</td>
</tr>
<tr>
<td>Internal physical signs</td>
<td>Physical signs which manifest themselves internally</td>
<td>Enlarged glands</td>
</tr>
<tr>
<td>Loss of appetite</td>
<td>A decrease in the amount of food the cow eats</td>
<td>Loss of appetite</td>
</tr>
<tr>
<td>Unusual excretions from upper body</td>
<td>The emission of fluids from the eyes, nose, ears or mouth which should not normally occur</td>
<td>Lachrymation, excessive salivation or nasal discharge</td>
</tr>
</tbody>
</table>

Table 3.1: Categories of Broad Symptoms used.
3.3. THE INFERENCE ENGINE

Johne’s Disease (Department of Agriculture, 2012c) all demonstrate how the disease is either spread directly from an infected animal, as the primary or a secondary method of infection, or how another animal residing nearby or under the same conditions as a sick one could pick up the same disease.

Should it have been confirmed in the queries presented in Section 3.3.1 that this cow had been in contact with another cow that had recently been diagnosed with a disease, a list of potential diseases is presented by the system. Any disease selected is deemed to be a more likely candidate in this particular diagnostic session.

3.3.5 Vaccination History

The InfoPak documents warn of an increased risk of the symptoms of some diseases appearing subsequent to the vaccination of an animal for a disease (Thomas, 2013) and of the increased likelihood of a disease occurring severely should the animal become infected with it before the immune system has had sufficient opportunity to build up defences against the disease (Turton, 1999).

If the user stated in the queries presented in Section 3.3.1 that the cow is regularly vaccinated, a list of diseases is presented and the user is requested to specify which disease the cow has been vaccinated for during the previous twelve months. Once the user has submitted the diseases for which the cow has been vaccinated, a list of these diseases is presented and the user is queried whether any of these vaccinations have been given within the past three weeks. Should this be the case, the vaccination could indicate an increased rather than reduced risk of the cow having the associated disease.

3.3.6 Initial Possible Disease List

At this stage the system determines which of the diseases it has knowledge about could potentially be the disease currently afflicting the cow. This preliminary decision is based on the broad symptoms discussed in Section 3.3.2 and the selections regarding immunity. Information about vaccination and other sick animals is also considered here.
3.3.7 Rulefire

Given the initial possible disease list, the system determines the groups of rules to be executed by the system in the next phase, where symptoms are queried. These rules, referred to as Rulefire in the development of the system, handle the execution of the rules of a particular disease by querying the user on specific symptoms and identifying matched symptoms.

3.3.8 Querying Symptoms

For each disease that has been identified as being a possible disease, the system queries the user on the actual existence of the various symptoms of this disease. Should a symptom have previously been queried with respect to a different disease, the symptom is not requested again; instead the result is taken from the previous query and applied to the disease currently under consideration.

3.3.9 Evaluation of Likelihood

The system calculates the likelihood of each of the possible diseases which were previously identified as possibly being the disease afflicting the cow, based on the symptoms that have been confirmed as manifested and other factors such as weather, time of the year, the cow's age, and the information about the vaccinations and other sick animals requested previously. See Section 3.4.4 for further details on how these calculations are done.

3.3.10 Differentiation Process

A process that attempts to differentiate between two diseases is employed when two diseases with similar symptoms are both deemed by the system to be highly likely. Ignoring symptoms that have been confirmed and which are common to the two diseases, the system determines how many symptoms unique to a particular disease are present. Should one disease have symptoms present that are unique to it and the other similar disease have none, the system will mark the former as being the most likely disease. Should symptoms unique to both diseases be present, the process will produce an undecided verdict between the two diseases, as it is not possible to accurately make a conclusion in this instance.
3.4. CONVERSION OF KNOWLEDGE TO RULES

The differentiation process does not attempt to assign importance to different symptoms, as the documents in the InfoPaks generally do not indicate levels of importance of symptoms, as explained in Section 3.3.2.1. It was deemed reasonable to assume that an animal with a particular disease would generally not have symptoms indicative of another disease, such as pale gums and red-coloured urine in the case of Redwater and Heartwater. Should this occur, it was decided that the system should rather return an inconclusive result and a best diagnosis with disclaimer attached, rather than potentially incorrectly eliminate the actual disease. Should this scenario occur the presence of a human veterinary expert able to conduct tests could be beneficial to the situation.

3.3.11 Diagnosis

The outcome of the diagnostic process is presented to the user, along with an option to view the logs of the diagnostic process (See Section 3.7).

3.4 Conversion of Knowledge to Rules

The following section outlines the decision-making process employed by the system to make a diagnosis.

3.4.1 Diseases Chosen

The diseases chosen for this proof-of-concept version of the CDS were Lumpy Skin Disease, Heartwater, Gallsickness and Asiatic Redwater. Lumpy Skin Disease is very different from the other three diseases, allowing for diagnosis sessions and the testing thereof involving a clearly identifiable disease. Heartwater, Gallsickness and Asiatic Redwater are all tick-borne and share many similar symptoms, with some symptoms unique to each of them. This allowed for diagnosis sessions, and the testing thereof, involving similar diseases and the testing of differentiation methods developed as part of the CDS.

3.4.2 Initial Possible Diseases

As a result of the CDS not being able to exclude a disease from contention for diagnosis purely because of a low number of symptoms present (Section 3.3.2.1), for each disease
capable of diagnosis, the CDS checks to see if any broad symptom which is a match for this disease has been identified as being present in the cow. Should this be the case, the disease is marked as a candidate for further investigation.

In this proof-of-concept stage no further inference is applied in the initial phase, because of the low number of diseases the system is currently capable of diagnosing. Factors such as the weather, time frame and vaccination are taken into account only in the diagnosis phase once further symptoms have been queried. This is to allow for the proper demonstration of the functioning of the rules in the diagnosis phase, which would be rendered irrelevant should it frequently be the case that only one disease makes it through the initial phase. The low number of possibly diagnosable diseases means that the number of symptoms queried from the user is not unnecessarily time-consuming. Should the system be expanded, further inference steps may need to be applied to reduce the number of diseases for which symptoms are queried from the user. Bearing in mind the importance of not excluding a disease merely because of a low number of symptoms present, the most likely method of implementation would be a phased approach, with additional inference applied to identify the most likely candidates. Diseases would be queried in phases, with the most likely candidates being queried first and the next phase of diseases only queried should a diagnosis with the required confidence not be made in the former phase.

Once the broad symptoms have all been checked, the total number of broad symptoms is obtained. Each disease is then checked to see how many of the broad symptoms which have been confirmed as being present apply to it. This number is calculated and, should it be one or greater, the disease will be considered for diagnosis and will have its symptoms queried. The weather conditions are also considered here as the risk of some diseases is increased with wet weather, while weather patterns are irrelevant to others. Should the weather not be a factor in the likelihood of the disease, the weather field is left unset, indicating its irrelevance.

Processing of vaccinations and infections of cows in contact with this cow also occurs at this stage.

### 3.4.3 Querying of Symptoms

Once the diseases that should be further investigated have been decided upon, each disease considered for diagnosis has its symptoms queried. The CDS works through each symptom the disease can display. The relevant rule will only fire if that symptom has not previously
$dis : PossibleDisease(code = "Lumpy Skin Disease", status = -1, possibility = -1)
$broad : BroadSymptoms()
...
then
  if ($broad.appetite) {
    modify ($dis) { incForbroad() };
  }
  if ($broad.external) {
    modify ($dis) { incForbroad() };
  }
...

Listing 3.1: Each broad symptom is checked. If the symptom applies to this particular disease, the count of broad symptoms is increased.

rule "Process vaccinations"
when
  $grouper : RuleGrouper(group = "START")
  $pd : PossibleDisease()
  $vac : Vaccination(disease = $pd.getCode(), processed = false)
then
  modify ($pd) { setVaccination($vac.getVaccination())};
  modify ($vac) { setProcessed(); };
end

Listing 3.2: The processing of vaccinations. The processing of infections in other cows is similar.
3.4. CONVERSION OF KNOWLEDGE TO RULES

rule "Gallsickness Pale to yellow gums"
when
  not Symptom(code == "Pale To Yellow Gums")
  not Processed(code == "SymptomQuery Pale To Yellow Gums")
$queries : Queries()$group : RuleGrouper(group == "Gallsickness−ADDSYM")
then
  insert(new Processed("SymptomQuery Pale To Yellow Gums"));
$queries.addSymptom("Pale to Yellow Gums", "Gums have become a pale to yellow colour");
end

Listing 3.3: The querying of a symptom. The Processed class is used to ensure that a symptom is only added once to the query class, while the ‘not Symptom’ checks to see if a symptom has previously been queried.

been queried. For example, both Gallsickness and Asiatic Redwater can cause a cow’s gums to become a pale to yellow colour. This symptom will only be queried once, and the result from the previous query will be used the second time the symptom is possible. Should the symptom not yet have been queried, the symptom is added to the list of symptoms to be queried for this disease.

Once all the symptoms have been checked to determine if they should be queried or not, the system moves to a rule which effectively ends the inference session. This ends the method which fires the rules in the servlet and allows the remaining code to be executed.

The Queries class receives the code used for the symptom (usually its name or a brief description) and the text which should be presented to the user to query the symptom. The class automatically adds the relevant HTML code which allows the form to display correctly and the servlet to retrieve the information from the form once it has been submitted by the user.

Once the form is submitted, the servlet extracts the parameters and processes these to determine if the symptoms have been confirmed. Symptom objects are then inserted by the servlet into the working memory, containing each symptom code and whether that symptom has been confirmed. The diagnostic session is then continued by triggering the command which fires rules in the knowledge session. The RuleGrouper class ensures that the knowledge session resumes firing rules from the correct point in the diagnostic session.

All symptoms that could be displayed by a disease are checked to determine if they are present. Should they be present, a count of the number of symptoms which are present for the disease is incremented. This process is performed for each disease which was identified in the previous phase (Section 3.4.2) as requiring investigation.
3.4. Evaluation Process

Following the conclusion of the querying of symptoms, the evaluation process begins. The number of symptoms which have been confirmed as being displayed by the cow but are not applicable to a particular disease is calculated and stored with that disease. This calculation is performed as the appearance of symptoms which are not generally produced by a particular disease could indicate a strong likelihood that it is not that disease which has afflicted the cow. The disease is also checked to determine if at least one symptom has been confirmed. Should this not be the case, the disease is immediately discarded from further consideration.

Following the above, the percentage of net symptoms which have been confirmed for a disease is calculated. "Net symptoms" refers to the number of symptoms that are applicable to the disease and have been confirmed, minus those confirmed symptoms not applicable to it. Net symptoms rather than matched symptoms is used since the presence of symptoms which are not particular to a disease could, especially if there are many, strongly indicate a low likelihood of that disease.

Later in the diagnostic process, diseases with a confidence value of lower than 60 are cut from consideration, as it is deemed that the confidence value would not be high enough to make a well-informed diagnosis. However, as explained in Section 3.3.2.1, a disease cannot be ruled out because of a low number of symptoms being present, hence the confidence level cannot be low purely because of a lack of matched symptoms. This is the reason that, following the calculation of this percentage for all diseases, the disease with the highest percentage is assigned a value of 100, which is now referred to as the confidence value of the disease occurring. Further modifications are then made to this value. The initial confidence level for all considered diseases is also raised by the same amount which the disease with the highest confidence had its confidence raised. The CDS can then make inferences regarding the confidence it has in a diagnosis without incorrectly dismissing the possibility of a disease purely because of a lack of symptoms being present. Examples of the value of the raising of the confidence levels to 100 are shown in Figures 3.6 and 3.7. In the former, only three symptoms are given for Asiatic Redwater, out of a possible 11. No symptoms which are not applicable to it are confirmed. Without the raising of confidences, the CDS rejects both Asiatic Redwater and Gallsickness owing to a low diagnostic score (as a result of the low confidences), despite Asiatic Redwater having no symptoms against it and a symptom unique to it confirmed. By raising the confidences, the rejection of a disease purely because of a low number of matched symptoms is avoided and the CDS is able to make a confident diagnosis.
3.4. CONVERSION OF KNOWLEDGE TO RULES

Figure 3.6: An example comparing a diagnosis made with the raising of confidence levels to one without this step performed.

Figure 3.7: An example showing the value of the “>= 60” condition.

Figure 3.7 shows that the condition requiring a diagnostic score to be above 60 is useful, as it eliminates the further consideration of diseases which are not reasonable potential diagnoses. In the example given, with all 11 symptoms attributable to Asiatic Redwater confirmed, Heartwater is eliminated from further consideration, owing to a low diagnostic score (given the likelihood value of one for all diseases in the example, the diagnostic scores and net percentage of matched symptoms would be identical).

The InfoPaks have been compiled by a variety of authors and do not refer to one another. It was therefore difficult to draw conclusions with regards to the likelihood of one disease versus another, and make inferences regarding the importance of one condition which increases the likelihood of one disease versus the importance of a likelihood-increasing condition for another disease. It was therefore decided not to have rules regarding one
3.4. CONVERSION OF KNOWLEDGE TO RULES

1 if ($pd.getVaccination() == 1) {
2     modify ($pd) { decRisk(0.3); }
3 }

Listing 3.4: The calculation of the likelihood factor.

disease versus another to assign a rank of probability but rather to assign a likelihood factor to each disease, with all diseases starting at a likelihood of one, and modify this risk according to the presence or lack thereof of certain conditions. The presence of weather conducive to a disease would increase the likelihood factor by 0.1, as would the time of year the diagnosis was being made falling over a period of increased likelihood for the disease. A recent vaccination for the disease and the presence of another cow with an infection for the disease is deemed to increase the likelihood of a disease significantly and both increase the likelihood factor by 0.2, while a vaccination within the past year lowers the risk factor by 0.3. The total likelihood factor for a disease is then calculated and its confidence is multiplied by this. The new value produced is referred to as a disease’s diagnostic score.

The diseases are checked subsequent to this and any disease with a diagnostic score of less than 60 is discarded from further evaluation, to prevent the system making a diagnosis where it cannot do so without a reasonable degree of confidence.

3.4.5 Differentiation of Similar Diseases

Following the calculation of the likelihood factor, the system checks whether two diseases which are known to be similar are in contention for diagnosis. Should this be the case, the system employs the differentiation technique described in Section 3.3.10. The disease differentiated against, if a result occurs, is removed from contention for diagnosis.

3.4.6 Diagnosis

The final step in the diagnosis procedure is to perform the final diagnosis. The disease left in contention with the highest confidence is diagnosed. Any other disease which remains in contention and has a diagnostic score similar to the disease that has been diagnosed ($<= 25$ points difference) or could not be successfully differentiated against is included in the diagnosis as a secondary diagnosis. The knowledge session is then terminated and the diagnosis is presented to the user.
3.4. CONVERSION OF KNOWLEDGE TO RULES

rule "Differentiate - Heartwater and Gallsickness"
when $group : RuleGrouper(group == "EVALUATE-5")
$pd1 : PossibleDisease(possibility == 4, code == "Heartwater")
$pd2 : PossibleDisease(possibility == 4, code == "Gallsickness")
$diff : Differentiator(dis1 == "Gallsickness", dis2 == "Heartwater", processed == false)
then
if ($diff.getResS().equals("Gallsickness")) {
    modify ($pd1) { setPossibility(3); };
}
if ($diff.getResS().equals("Heartwater")) {
    modify ($pd2) { setPossibility(3); };
}
modify ($diff) { setProcessed(true); }
end

Listing 3.5: The differentiation between Heartwater and Gallsickness, if required. The Differentiator contains the result ($diff.getResS()) as a result of rules previously fired.

<table>
<thead>
<tr>
<th>Confidence</th>
<th>Only one disease diagnosed</th>
<th>More than one possible disease diagnosed</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;100</td>
<td>Very high</td>
<td>Medium</td>
</tr>
<tr>
<td>91 - 100</td>
<td>High</td>
<td>Medium</td>
</tr>
<tr>
<td>71 - 90</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>60 - 70</td>
<td>Medium</td>
<td>Low</td>
</tr>
</tbody>
</table>

Table 3.2: The rating system used to convert the calculated confidence score to a textual descriptor.

As the confidence score calculated by the CDS is an internal metric and cannot be measured in standardised units, a rating system is applied which assigns a textual descriptor to the confidence score, which is then presented to the user instead. This textual description indicates the degree of confidence the system has in the diagnosis it has made. The rating system is shown in Table 3.2. The figures were chosen based on the figures used to determine the likelihood factor of a disease. As the disease with the highest number of matched symptoms is set to 100 confidence, this indicates a “high” confidence, while a diagnostic score of over 100 would indicate, for a disease which received a 100 confidence score, that the likelihood factor was above the base level of one. A score of over 100 was therefore assigned a “very high” descriptor. As an annual vaccination reduces the likelihood by 0.3, to 0.7, assuming there are no other factors, 70 was used as the cut-off point for a diagnostic score to be labelled as “high” confidence. If the CDS returned more than one result, the highest confidence descriptor which could be assigned was “medium”, as it was felt that the presence of doubt indicated that a “high” diagnostic confidence should not be indicated.
3.5 HTTP Servlet

After an investigation and an attempt at incorporating Drools Execution Server as the interface between the knowledge base and working memory and the UI, a decision was made to develop a custom servlet to perform this function. This allows for more control over the communication process and the ability to tailor the servlet to the unique operation of this system, allowing for easier updating of the UI. The servlet extends Java’s HTTPServlet class and implements a select subset of that class’ methods.

The constructor method for the servlet performs the setup to allow the knowledge session to be conducted by building the knowledge base from the various DRL files. A new knowledge session is then created from the knowledge base. It is this knowledge session that will be invoked to perform the diagnosis.

A text file containing a list of the diseases and the total number of symptoms that are associated with each disease is read in and this information stored in an ArrayList. This data is used to insert the PossibleDisease classes into the working memory and to produce the HTML forms for the initial queries about immunity, vaccination and possible diseases in other cows. The text file is used to make the addition of further diseases to the system a user-friendly process.

The doPost() method receives post requests from the browser to the servlet and handles all the processing following the instantiation of the object. The doPost() method has two parameters, a request and response, from which it can receive parameters and return HTML code, respectively. Included in the request variable is the URL of the request, which is used as the starting-point for the processing of the request. The path is extracted and, dependent on this, parameters are obtained and inserted into the working memory, rules are fired and the generated response is extracted from the working memory and sent to the UI.

Each form sent to the UI for the user to interact with submits back to the servlet with a URL unique to the function for which this form has been used. For example, the form that requests the user to provide information about vaccinations the cow has received submits to a “/vaccinations” URL, while the form which queries the user for symptoms specific to a disease submits to “/symptomqueries” (See Listing 3.6).

The first time the user submits information, which is after filling in basic information about the cow, the system instantiates various classes that will be used later in the
protected void doPost(HttpServletRequest request, HttpServletResponse response) {
    String path = request.getPathInfo();
    if (path.equals("/initial")) {
        ...
    } else if (path.equals("/broadsymptoms")) {
        ...
    }
}

Listing 3.6: Extraction of the URL path and the processing that occurs dependent on this.

else if (path.equals("/broadsymptoms")) {
    //PROCESS REQUEST
    BroadSymptoms bs = HelperFunctions.broadRequest(request);
    ks.ksession.insert(bs);
    //GENERATE RESPONSE
    if (ks.cow.isPrevious()) { //Previously survived a disease
        result = HelperFunctions.getImmunityQueryHTML(ks.diseases);
    } else if (ks.cow.isHerd()) { //Another cow is sick
        result = HelperFunctions.getHerdQueryHTML(ks.diseases);
    } else if (ks.cow.isVaccinated()) { //The cow is vaccinated
        result = HelperFunctions.getVaccineQueryHTML(ks.diseases);
    } else { //Start the diagnostic session
        ks.ksession.fireAllRules();
        result = HelperFunctions.getQueriesHTML(ks.queriesHandle.toString());
    }
}

Listing 3.7: The processing of the request followed by the generation of the response HTML code, which is stored in result.

diagnostic session to extract information and inserts them into working memory. They are inserted using a FactHandle\(^5\), which allows an object outside the working memory, in this case the servlet, to later update this object or obtain information from it.

In the initial phase where items such as immunity, vaccinations and broad symptoms are queried, facts are inserted into the working memory and a response is generated without firing any rules. Listing 3.7 shows the code used for this. The response here is dependent on the answers given to the general information queries. Should the user have indicated that the cow has previously survived a disease, the response will be the form requesting information about which disease(s) was involved. The form will submit to "'/immunity'" and the same if-statement will be used to generate the next response, this time without the ks.cow.isPrevious() condition.

\(^5\)When an object is inserted into working memory, a FactHandle keeps a "handle" on this object, allowing it to be accessed from outside the working memory.
else if (path.equals("/symptomquery")) {
    //PROCESS REQUEST
    Get from the working memory a list of the symptoms queried this time
    For each symptom
        Get its parameter from the HTML request
        Insert the Symptom into working memory, specify whether confirmed or not
    //FIRE RULES
    Update the RuleGrouper in working memory to specify where in the knowledge base to resume firing rules from
    Fire the rules
    //PRODUCE RESPONSE
    If there are still symptoms to be queried
        Get the form querying these symptoms from working memory
    Else
        Get the text containing the diagnosis from working memory
}

Listing 3.8: The processing of symptoms which have been queried.

Following the commencement of the diagnostic session, the symptoms are queried and placed into the working memory. Each symptom is extracted and a Symptom object created. Subsequent to this, the knowledge session is restarted. After the firing of the rules has halted, another disease’s symptoms will be queried or a result will be available, depending on the depth of progression into the diagnosis. The pseudo-code for this procedure is shown in Listing 3.8.

3.6 Mobile Interface

3.6.1 Choice of Interface

The interface for the CDS is presented in a mobile website. This decision was made to avoid tying the system to one particular platform. As the primary function of the interface for this system is the input of data through selection from pre-defined lists of choices, the advantages of native smartphone application development would be nullified for the purposes of this project. No interaction with system components or data-sharing with other applications is required, while the nature of the forms being generated means that there is no real benefit in the reduction of the size of the communication by drawing on native UI components. The use of a native application would therefore restrict the use of the system while deriving no useful benefit in return.
The CDS can operate on any mobile web browser supporting HTML and JavaScript. The use of basic elements was done to allow the UI to function correctly on low-end mobile phones without powerful processors or large screen resolutions. Information about the testing of the UI can be found in Section 4.3.

### 3.6.2 Basic Structure of the Interface

The overall structure of the interface is a JavaServer Page which consists of a header and footer, which serve a purely cosmetic purpose, and an HTML `<div>` tag. The `<div>` tag’s content is updated throughout the diagnostic session with HTML code received from the HTTP servlet.

The structure of the interface is built using HTML, with the addition of JQuery Mobile components and JavaScript scripts which serve to provide extended functionality and components. A key portion of the functionality of the interface is a JQuery script which facilitates the dynamic updating of the interface using AJAX. A stylesheet provided by JQuery Mobile manages the layout and style of the page, making for a more presentable interface.

### 3.6.3 Interaction using Dynamic Forms

The HTML code containing the skeleton of all the forms used in the system is shown in Listing 3.9. The first and last lines contain the `<div>` tag. This tag is constant but
3.6. MOBILE INTERFACE

Listing 3.9: The HTML code used for the forms.

```html
1  <div id="content" data-role="content">
2      <script>
3          // JQuery script shown in Listing 1.10
4      </script>
5      <form id="cowform" method="post" action="CowVetServlet/initial"/>
6          // Form components as applicable
7          <input type="submit" value="Submit" />
8      </form>
9  </div>  <!-- /content -->
```

The JQuery script in Listing 3.10 binds the submit action of the “cowform” to the function which is defined inside the script. The function posts the form in formation to the action URL specified in the form and receives the resultant response in the variable `responseText`. This text is then set as the HTML for the `<div>` tag “content”, effectively replacing the form which triggered the submit action, with a new one. This script is placed inside the `<div>` tag as it is returned along with the new form on each update, as this is required for it to function correctly. The same reason applies for the “submit” button being resent on each update, despite appearing on each page.

The use of AJAX allows the `div` tag’s content to be updated after each form submission without refreshing the entire web page and potentially resetting the servlet, losing the progress of the invoked knowledge session.

3.6.4 Index Page

The home page for the system (index.jsp) is the only page the user sees that is not dynamically produced. This page is statically coded and is served up when the user enters the site. It contains the fields for the user to fill in the basic queries, such as whether the cow is vaccinated, and information about its age. The servlet is invoked only when the “submit” button is clicked on this page.
Listing 3.10: The JQuery script used to provide the AJAX functionality to update the web page.

Figure 3.9: The index page which greets the user when they visit the system.
3.6. MOBILE INTERFACE

Listing 3.11: The code used to generate the HTML to query which disease(s) the cow has previously survived.

3.6.5 Initial Queries

The initial queries, such as the questions about immunity, do not rely on the working memory but are partially pre-configured HTML forms. As these queries ask the user a question and require them to select one or more diseases from a list of diseases the only variable is the list of diseases which the system can support. Each disease is therefore added as a checkbox option in the list, which is then placed in the form. Listing 3.11 demonstrates this process.

The queries about vaccination and other cows which this cow may potentially have come into contact with are almost identical. The query about which vaccinations have occurred recently, however, only lists those vaccinations which have been selected as having occurred, if any.

3.6.6 Symptom Queries

The Queries class generates HTML code to query a batch of symptoms in the UI. Listing 3.11 shows how the code used for the symptom in the system and text which should be used to query the system is added to this object for each symptom which is to be queried. The query text is padded with HTML to form its specific checkbox in the list; this process is shown in Listing 3.12.

Drools provides support for obtaining an Object from the knowledge session’s working memory only. A Queries object cannot be extracted from the working memory and...
Figure 3.10: The broad symptoms query form.

```java
public void addSymptom(String s, String q) {
    this.symptoms.add(s);
    int num = this.symptoms.size();
    this.queryString += "<input type="checkbox" name="checkbox-" + num + " id="checkbox-" + num + ">
    " + q + "</label>
    ";
}
```

Listing 3.12: The adding of a symptom into the Queries class.
used in the servlet code, nor can one of its custom methods return the relevant classes or information; the only methods which can be called are those that exist in the Object interface itself. The Queries' toString() method was therefore overridden to provide the relevant information. The queries text with padded HTML is further padded with the HTML code to produce the form. This is demonstrated in Listing 3.13. Although not shown in the listing, the method also returns a list of the symptom codes. This allows the servlet to construct Symptom objects when it reads the parameters from the result of this form submission. String manipulation is used by the servlet to obtain the correct portions of information from the toString() method.

3.6.7 Diagnosis

The Diagnosis class overrides its toString() method and returns its information in a manner very similar to that of the Queries class (Section 3.6.6). The disease diagnosed is printed out, along with its diagnostic score. A descriptor indicating the level of confidence the CDS has in its diagnosis is provided, along with information about the disease and what steps should be taken following this diagnosis.

This information is read in from text files which are hosted on the server. The separation of this information from the code allows for easy updating of the text should it need to be changed in future, in line with the secondary aim of simplifying the process of adding disease information for a non-technically skilled user.

3.7 Logging Functionality

The CDS provides a basic logging facility to enable someone to examine the decision-making process followed while reaching its conclusion. This facility is provided to fulfil
Figure 3.11: An example of a form querying some symptoms.

Figure 3.12: The response after a diagnosis has been made.
3.8. IMPLEMENTATION SPECIFICS

the requirement that an expert system should be able to explain its logic, and should not be a black box with no ability to see the inner workings. This is particularly important where the life of a human-being or animal is at stake.

3.7.1 Logging During the Diagnostic Session

Events which are pertinent to the eventual diagnosis made by the CDS are logged. During the development process the firing of any rule was logged but this was scaled back to log only the information a veterinarian would find pertinent should they wish to verify the CDS’s diagnosis or decision-making process. Items such as the matching of a symptom with a disease, the processing of a vaccination, contact with another sick cow or immunity for a disease, failed or successful differentiation between two diseases and the calculations made in the decision making process are logged. The logs are saved to a text file at the conclusion of the diagnostic session.

3.7.2 Viewing the Logs

When the diagnosis is presented to the user at the conclusion of the diagnostic session an option to view the logs for that session is provided. The log text file opens in a separate tab upon the selection of this option.

3.8 Implementation Specifics

The CDS makes use of various classes which are used in both the servlet and knowledge base components. Some classes are used in the core operation of the knowledge base while others facilitate communication between the servlet and the knowledge base. Some classes also exist to perform minor functions which ensure the smooth running of the system.

3.8.1 Initial Phases and Setup

3.8.1.1 BroadSymptoms

This class stores which of the broad symptoms outlined in Section 3.3.2 were marked as present in the cow being diagnosed. The weather conditions are also stored.
3.8. IMPLEMENTATION SPECIFICS

3.8.1.2 Cow

The information gathered about the cow include whether the cow is regularly vaccinated, has previously survived a disease or whether another cow has recently been diagnosed with a disease. This information is used by the CowVetServlet class (Section 3.8.4.1) to determine which page to display on the UI.

3.8.1.3 HerdInfection, Immunity and Vaccination

The HerdInfection, Immunity and Vaccination classes store information about diseases which may have been diagnosed in animals which the cow being diagnosed could have come into contact with, may have previously survived, or been vaccinated against, respectively. The Vaccination class also stores whether this vaccination has recently been administered.

3.8.2 Symptoms and Diagnosis

3.8.2.1 Diagnosis

This class holds the outcome of the diagnostic session. The disease deemed most likely to be present in the cow and any diseases which also demonstrate a strong possibility of being present and could not be ruled out through the differentiation process (where applicable) are stored here.

3.8.2.2 DiagnosisHelper

This assistant class is used in the evaluation process after symptoms have been queried to hold the outcomes of calculations which are required in rules other than the one in which they have been generated.

3.8.2.3 Differentiator

The differentiation process between two diseases is controlled by this class, which looks for symptoms that are present and are unique to one of the two diseases being considered. The outcome of this process is stored and is available to any rules requiring it.
3.8. IMPLEMENTATION SPECIFICS

3.8.2.4 PossibleDisease

Information about diseases which the system is reasoning over is contained in this class. The number of symptoms applicable to a disease and the number of symptoms which have occurred but do not correspond to that disease are stored, along with information regarding factors such as vaccinations and time frame, which are used to generate the overall risk factor for the disease. The class contains several fields which are used by the rules to determine the state of the disease and whether it is still being considered as a disease that could potentially be diagnosed in this session.

3.8.2.5 SuspectedDiseases

An array of diseases deemed likely to be diagnosed is stored here for final evaluation. The class is a helper class and is used for the management of calculations.

3.8.3 Communication

3.8.3.1 Next

This class serves to inform the servlet whether it should extract code for symptom queries for display in a form or diagnosis information from the working memory.

3.8.3.2 Queries

The class contains a list of all symptoms which are to be queried next and the text which should be used to query them. Its toString() method wraps HTML around this information and allows it to be displayed in a form in the UI, as discussed in Section 3.6.6.

3.8.4 Servlet Classes

3.8.4.1 CowVetServlet

This class forms the Servlet which is responsible for the communication between the UI and the knowledge base. The initial setup of the knowledge base and knowledge session is
performed, followed by the processing of parameters and responses provided by the user and the insertion of these into working memory. The results of the various phases of the inference process, including further queries for the user, are extracted from the working memory and sent to the UI for completion.

### 3.8.4.2 HelperFunctions

This static class is never instantiated and is primarily for the abstraction of functions in the CowVetServlet class to improve code readability.

### 3.8.4.3 KnowledgeSession

The KnowledgeSession class holds the working memory of this particular diagnostic session. This working memory has been generated from the knowledge base in the CowVetServlet class. Various FactHandles are present which allow information to be extracted from the working memory and processed by the CowVetServlet class, which can send the information on to the UI. It is also used as a convenient place to store information about the diseases the system can diagnose and the vaccinations the cow has received for the generation of the interface for queries in the initial phases of the diagnostic process.

### 3.8.5 Rule Control

#### 3.8.5.1 Processed

This class serves to mark when rules which should only fire once, have fired. Although Drools provides native functionality to fulfil this by marking rules as lock-on-active, this was found during the development process to be temperamental due to agenda-groups not being used in the system (Section 3.8.5.2). This custom solution was utilised instead.

#### 3.8.5.2 RuleGrouper

The RuleGrouper class groups rules according to a purpose or disease and controls which set of rules should be examined by the inference engine. For instance, during the initial
phase the RuleGrouper ensures that only rules which are responsible for determining initial possible diseases can be fired and when a particular disease is being examined, only rules relevant to that disease are available to be fired.

Drools provides native functionality for this function, called agenda-groups, but the reliability of this functionality was found to be temperamental. This was discovered to be a known bug in the version of Drools being used in the development of the system\(^6\) and so the custom RuleGrouper class was utilised to provide this functionality.

### 3.8.6 Logging Functionality

#### 3.8.6.1 Logger

The class contains an ArrayList of the LoggerItems which provide the logging functionality for the CDS. The ArrayList was placed in a class before being placed into working memory to facilitate development and code legibility. The Logger class’ contents are written to a text file at the culmination of the diagnostic session.

#### 3.8.6.2 LoggerItem

Each LoggerItem object contains the information pertaining to one event logged.

### 3.9 Summary

This chapter dealt with the specifics of the design and implementation of the CDS. An overview of the CDS was first presented. The knowledge source and acquisition methods were discussed and an explanation provided of the process for converting this knowledge into rules, as well as an explanation of the diagnostic process followed. The development of the HTTP Servlet and UI was discussed, together with the logging functionality. Finally, specifics of the implementation of the CDS were briefly explained.

\(^6\)https://bugzilla.redhat.com/show_bug.cgi?id=1016727
Chapter 4

Results and Analysis

In this chapter the CDS is tested with a variety of cases and the results of tests are analysed and discussed. A reference table indicating which symptoms can occur with which diseases is first provided. Finally, an explanation regarding the testing of the UI is provided and the overall functionality of the CDS is discussed.

4.1 Symptom Reference Table

Table 4.1 is provided as a reference to avoid duplication of information in the results. It shows which symptoms apply to each particular disease in the system. Symptoms which occur are marked with an “X”. Note that lack of appetite and a drop in milk production are not included in the list of individual symptoms in the results presentation in Section 4.2, but are included in broad symptoms only, to avoid redundancy.

The current month is automatically drawn by the CDS from the current system date; however, for the purposes of testing, this was manually set in the code in some instances.

4.2 Results of Diagnostic Tests

The testing of the system was necessarily limited by the scope of the project, and no real-life cases were presented to the system. Rather, a series of hypothetical cases, some more realistic than others, was presented to the CDS. These cases were designed to test the
<table>
<thead>
<tr>
<th>Symptom</th>
<th>Asiatic Redwater</th>
<th>Gallsickness</th>
<th>Heartwater</th>
<th>Lumpy Skin Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lack of appetite</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Blinking when tapped</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chewing movements</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Constipation</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Convulsions</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Depression</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Emaciation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enlarged lymph nodes</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Fever</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Frequent coughing</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Lachrymation</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Drop in milk production</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Nasal discharge</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Nervous signs</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Pale to yellow eyes</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Pale to yellow gums</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Red urine</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Salivation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Skin nodules with ulcers</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Swollen legs with sores</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Weakness</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Walking unusually</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

Table 4.1: Symptoms and the diseases to which they apply.
functioning of various aspects of the diagnostic process and the viability and correctness of the inference logic built into it. Each result is presented with an introduction, a full account of the symptoms and other conditions used, the results of key calculations made by the system and the diagnosis presented. The diagnosis of each test case is then commented on.

Note that in these results “herd infection” refers to the diagnosis of a disease in another cow which the cow under investigation could have come into contact with, as this was the term used during the development of the CDS.

### 4.2.1 Basic Diagnosis Functionality

The first test served to demonstrate the basic functionality of the CDS and that the basic operation performed as intended. All symptoms which could be attributed to LSD are confirmed, while no symptoms of another disease (not shared by LSD) are. No immunity, herd infection or vaccination was marked as present. The variables used are shown in Table 4.2.

| Symptoms: | • Emaciation  
|           | • Enlarged lymph nodes  
|           | • Fever  
|           | • Frequent coughing  
|           | • Lachrymation  
|           | • Nasal discharge  
|           | • Salivation  
|           | • Skin nodules with ulcers  
|           | • Swollen legs with sores  
| Broad Symptoms: | • Drop in milk production  
|                | • External physical signs  
|                | • Fever and coughing  
|                | • Internal physical signs  
|                | • Loss of appetite  
|                | • Unusual excretions  
| Immunity: | None  
| Herd Infections: | None  
| Vaccinations: | None  
| Wet Weather: | No  
| Month: | October  
| Cow Age: | 5 years 0 months  
| Diseases for Investigation: | All  
| Confidence (Adjusted Confidence): | • Astatoch Redwater: -70 (Discarded)  
|                                    | • Babcock: -78 (Discarded)  
|                                    | • Heartwater: -78 (Discarded)  
|                                    | • LSD: 91 (100)  
| Likelihood: | LSD: 1.0  
| Score: | LSD: 100  
| Diagnosis: | Lumpy Skin Disease (High)  

Table 4.2: Basic diagnosis functionality.
### Table 4.3: Vaccination does not override an overwhelming presence of symptoms.

<table>
<thead>
<tr>
<th>Symptoms</th>
<th>Broads Symptoms</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Emaciation</td>
<td>• Drop in milk production</td>
</tr>
<tr>
<td>• Enlarged lymph nodes</td>
<td>• External physical signs</td>
</tr>
<tr>
<td>• Fever</td>
<td>• Fever and coughing</td>
</tr>
<tr>
<td>• Frequent coughing</td>
<td>• Internal physical signs</td>
</tr>
<tr>
<td>• Lachrymation</td>
<td>• Loss of appetite</td>
</tr>
<tr>
<td>• Nasal discharge</td>
<td>• Unusual excretions</td>
</tr>
<tr>
<td>• Salivation</td>
<td></td>
</tr>
<tr>
<td>• Skin nodules with ulcers</td>
<td></td>
</tr>
<tr>
<td>• Swollen legs with sores</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Immunity</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Herd Infections</td>
<td>Heartwater</td>
</tr>
<tr>
<td>Vaccinations</td>
<td>LSD (not recent)</td>
</tr>
<tr>
<td>Wet Weather</td>
<td>No</td>
</tr>
<tr>
<td>Month</td>
<td>October</td>
</tr>
<tr>
<td>Cow Age</td>
<td>5 years 0 months</td>
</tr>
<tr>
<td>Diseases for Investigation</td>
<td>All</td>
</tr>
<tr>
<td>Confidence (Adjusted Confidence)</td>
<td>• Aplastic Redwater: 70 (Discarded)</td>
</tr>
<tr>
<td></td>
<td>• Gall sickness: -78 (Discarded)</td>
</tr>
<tr>
<td></td>
<td>• Heartwater: 78 (Discarded)</td>
</tr>
<tr>
<td></td>
<td>• LSD: 100 (100)</td>
</tr>
<tr>
<td>Likelihood</td>
<td>LSD: 0.70</td>
</tr>
<tr>
<td>Diagnostic Score</td>
<td>LSD: 70</td>
</tr>
<tr>
<td>Diagnosis</td>
<td>Lumpy Skin Disease (High)</td>
</tr>
</tbody>
</table>

The CDS performed as expected, diagnosing LSD with a high confidence. Few symptoms for the other diseases would have matched, with LSD having such dissimilar symptoms to them, and so they were easily discounted from the diagnostic session, leaving LSD the clear candidate for diagnosis.

#### 4.2.2 Vaccination does not Override an Overwhelming Presence of Symptoms

This test, also simple, tests to see whether the overwhelming presence of symptoms of a disease is not cancelled out by the vaccination for it in conjunction with the infection of another animal with a different disease that has few matched symptoms in this diagnostic session. The test variables are presented in Table 4.3.

The system presents LSD as the diagnosis with lowered confidence (diagnostic score = 70) as a result of the vaccination that the cow has received for LSD. The number of symptoms indicated for LSD meant that it would be improbable that Heartwater was the disease present, despite LSD’s vaccination and the Heartwater infection in another cow.
4.2. RESULTS OF DIAGNOSTIC TESTS

4.2.3 Successful Differentiation: Asiatic Redwater and Heartwater

Differentiation was tested by presenting the CDS with five shared symptoms for Heartwater and Asiatic Redwater, along with one symptom specific to Asiatic Redwater. Table 4.4 shows the inputs used for this test.

The herd infection of Gallsickness and the recent vaccination were given to attempt to throw the system off by over-favouring them, owing to the increased risk which they would be associated with. Although the likelihood factor calculated by the CDS was higher for Heartwater and Gallsickness than for Asiatic Redwater, the number of symptoms for Gallsickness was not high, rendering it discarded by the CDS owing to its negative net number of matched symptoms. The higher likelihood for Heartwater was invalidated by the differentiation mechanism when examining it and Asiatic Redwater. Should Gallsickness’s confirmed symptoms have been higher, a similar differentiation process would have confirmed Asiatic Redwater above it as well.

| Symptoms | • Convulsions  
| • Depression  
| • Nervous signs  
| • Red urine  
| • Behavioural changes  
| • Digestive and urinary  
| • Drop in milk production  
| • Loss of appetite  
| Immunity | None  
| Herd Infections | None  
| Vaccinations | Heartwater (recent)  
| Asiatic Redwater (not recent)  
| Wet Weather | Yes  
| December  
| Month |  
| Cow Age | 5 years 0 months  
| Diseases for Investigation | All  
| Confidence (Adjusted Confidence) | • Asiatic Redwater: 40 (96)  
| • Gallsickness: 0 (Discarded)  
| • Heartwater: 44 (100)  
| • LSD: -38 (Discarded)  
| Likelihood | • Asiatic Redwater: 0.7  
| • Heartwater: 1.2  
| Diagnostic Score | • Asiatic Redwater: 67  
| • Heartwater: 120  
| Diagnosis | • Asiatic Redwater over Heartwater  

Table 4.4: Successful differentiation: Asiatic Redwater and Heartwater.
4.2. RESULTS OF DIAGNOSTIC TESTS

4.2.4 Unsuccessful Differentiation: Heartwater and Asiatic Redwater

The same test as in Section 4.2.3 was run, with the addition of a symptom unique to Heartwater. Table 4.5 applies to this test.

In this test the CDS was unable to differentiate between Heartwater and Asiatic Redwater; the CDS therefore returned an unqualified result. Heartwater had a slightly higher confidence than Asiatic Redwater, as a result of the recent vaccination. It is therefore reasonable for the system to state that considering the two diseases, Heartwater was more likely, provided it stated the possibility of Asiatic Redwater and its inability to make a confident differentiation.

4.2.5 Unsuccessful Differentiation: Gallsickness and Asiatic Redwater

This test was designed to explore the diagnosis made by the CDS when faced with the choice between two diseases it cannot differentiate between, but with one having a significantly lowered likelihood owing to having been vaccinated against. Five shared symptoms
4.2. RESULTS OF DIAGNOSTIC TESTS

Table 4.6: Unsuccessful differentiation: Gallsickness and Asiatic Redwater.

for Gallsickness and Asiatic Redwater were input, along with one unique symptom for each disease, and it was stated that a vaccine for Asiatic Redwater had been given in the past year. Table 4.6 applies to this test.

In this test the system was again unable to differentiate between two diseases, this time Gallsickness and Asiatic Redwater. In this instance, one of the diseases, Asiatic Redwater, had a significantly reduced likelihood as a result of the cow receiving a vaccination for it, and Gallsickness achieved a higher diagnostic score as a result. The CDS thus diagnosed Gallsickness as the most likely disease. Owing to Asiatic Redwater not having been successfully differentiated against, it is also included in the diagnosis, but with a lower confidence.

### 4.2.6 Unsuccessful Diagnosis: Insufficiently High Confidence

This test aimed to test that the CDS would not make a diagnosis where it could not do so without a reasonable degree of confidence. A mix of symptoms that would ensure that each disease had a few matched symptoms was marked as present. The variables used for this test are shown in Table 4.7.
4.2. RESULTS OF DIAGNOSTIC TESTS

Table 4.7: Unsuccessful diagnosis: Insufficiently high confidence.

The CDS was unable to make a diagnosis as each disease presented some symptoms, as each disease achieved a negative net number of matched symptoms. No disease could reasonably be favoured as a result of this, and the CDS was unable to return a result with any reasonable confidence.

4.2.7 Age as a Factor in the Diagnosis

In the following test, four symptoms specific to LSD were provided, along with two symptoms common among the three tick-borne diseases, but not applicable to LSD. Red urine was also given as a symptom, ranking Asiatic Redwater slightly higher among those diseases spread by ticks. This served to place LSD as having the highest net symptoms, with the least symptoms counting against it. The object of the test was to determine the effect of age, with an age being giving where the tick-borne diseases would have a slightly increased likelihood, as a result of older cattle being more susceptible to them. Table 4.8 applies to this test.

Despite the tick-borne diseases presenting a slightly higher likelihood of being present, none of them presented a positive net number of matched symptoms, with all of them having less symptoms matched than the number of symptoms applying to them that were
4.2. RESULTS OF DIAGNOSTIC TESTS

Table 4.8: Age as a factor in the diagnosis.

<table>
<thead>
<tr>
<th>Symptoms</th>
<th>Age 12 years 5 months</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depression</td>
<td></td>
</tr>
<tr>
<td>Nasal discharge</td>
<td></td>
</tr>
<tr>
<td>Red urine</td>
<td></td>
</tr>
<tr>
<td>Salivation</td>
<td></td>
</tr>
<tr>
<td>Skin nodules with ulcers</td>
<td></td>
</tr>
<tr>
<td>Swollen legs with sores</td>
<td></td>
</tr>
<tr>
<td>Weakness</td>
<td></td>
</tr>
<tr>
<td>Broad Symptoms</td>
<td></td>
</tr>
<tr>
<td>Behavioural changes</td>
<td></td>
</tr>
<tr>
<td>Digestive and urinary</td>
<td></td>
</tr>
<tr>
<td>External physical signs</td>
<td></td>
</tr>
<tr>
<td>Internal physical signs</td>
<td></td>
</tr>
<tr>
<td>Unusual excretions</td>
<td></td>
</tr>
<tr>
<td>Immunity</td>
<td>None</td>
</tr>
<tr>
<td>Herd Infections</td>
<td>None</td>
</tr>
<tr>
<td>Vaccinations</td>
<td>None</td>
</tr>
<tr>
<td>Wet Weather</td>
<td>No</td>
</tr>
<tr>
<td>Month</td>
<td>October</td>
</tr>
<tr>
<td>Cow Age</td>
<td>12 years 5 months</td>
</tr>
<tr>
<td>Diseases for Investigation</td>
<td>All</td>
</tr>
<tr>
<td>Asiatic Lethargy</td>
<td>-10 (Discarded)</td>
</tr>
<tr>
<td>Gall sickness</td>
<td>-33 (Discarded)</td>
</tr>
<tr>
<td>Heartwater</td>
<td>-33 (Discarded)</td>
</tr>
<tr>
<td>LSD</td>
<td>9 (100)</td>
</tr>
<tr>
<td>Likelihood</td>
<td>LSD: 1.0</td>
</tr>
<tr>
<td>Diagnostic Score</td>
<td>LSD: 100</td>
</tr>
<tr>
<td>Differentiation</td>
<td>n/a</td>
</tr>
<tr>
<td>Diagnosis</td>
<td>Lumpy Skin Disease (High)</td>
</tr>
</tbody>
</table>

present in the cow. LSD was therefore diagnosed. As LSD does not have a particular age-range for susceptibility, and cattle of all ages face an equal chance of being infected by this disease, all other factors being equal, this diagnosis could be seen as correct for the CDS to make.

In a case such as this, the CDS should perhaps have presented a reduced likelihood, as LSD has a number of symptoms against it. A threshold of a number of symptoms against a disease could be specified, after consultation with an expert, before a reduced confidence is shown, or a diagnosis for that disease is not made.

4.2.8 Similar Symptoms for Non-Similar Diseases: No Diagnosis

The differentiation facility was provided in the CDS to differentiate between two diseases which have many similar symptoms that could often cause confusion as to the correct diagnosis. This is not provided for in diseases which are not similar as the normal course of events in the diagnostic process would automatically do this: a disease with many more symptoms than the other would automatically be favoured. Should two diseases which are not similar be close together in terms of overlapping symptoms it could be considered
4.2. RESULTS OF DIAGNOSTIC TESTS

Table 4.9: Similar symptoms for non-similar diseases: no diagnosis.

<table>
<thead>
<tr>
<th>Symptoms</th>
<th>Broad Symptoms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blinking when tapped</td>
<td>Behavioural changes</td>
</tr>
<tr>
<td>Chewing movements</td>
<td>External physical signs</td>
</tr>
<tr>
<td>Emaciation</td>
<td>Fever and cough</td>
</tr>
<tr>
<td>Enlarged lymph nodes</td>
<td>Internal physical signs</td>
</tr>
<tr>
<td>Frequent coughing</td>
<td>Unusual excretions</td>
</tr>
<tr>
<td>Lachrymation</td>
<td></td>
</tr>
<tr>
<td>Nasal discharge</td>
<td></td>
</tr>
<tr>
<td>Nervous signs</td>
<td></td>
</tr>
<tr>
<td>Walking unusually</td>
<td></td>
</tr>
<tr>
<td>Weakness</td>
<td></td>
</tr>
</tbody>
</table>

| Immunity                  | None                            |
| Herd Infections           | None                            |
| Vaccinations              | None                            |
| Wet Weather               | No                              |
| Month                     | October                         |
| Cow Age                   | 5 years 0 months                |
| Diseases for Investigation| All                             |
| Confidence (Adjusted Confidence) |                             |
| Asomatic Redwater: -60 (Discarded) |                     |
| Gallsickness: -89 (Discarded) |                      |
| Heartwater: -0 (Discarded) |                           |
| LSD: 0 (Discarded)        |                                |
| Likelihood                | n/a                             |
| Diagnostic Score          | n/a                             |
| Differentiation           | n/a                             |
| Diagnosis                 | No diagnosis was made           |

Incorrect for the system to make a confident diagnosis, as this would not entail only one symptom for each disease not matching, as might be the case between similar diseases such as Gall sickness and Heartwater, but rather a large number of symptoms not matching. Five symptoms unique to LSD were given and five symptoms unique to Heartwater, when compared to LSD only and no other tick-borne diseases. Table 4.9 applies to this test case.

The CDS was unable to make a diagnosis, as both of the most likely candidates for diagnosis presented an equal number of symptoms. This can be considered the correct action for the CDS as any diagnosis made would have been made with a very low confidence.

4.2.9 Similar Symptoms for Non-Similar Diseases: Diagnosis Made

A very similar set of inputs was used in this test, shown in Table 4.10, as in the previous one, with the addition of one extra symptom for LSD, salivation.

In this case, the addition of a symptom for LSD resulted in the CDS making a confident diagnosis for LSD. This however raised a potential flag regarding the process used in the
beginning of the evaluation phase for the passing and discarding of diseases to the next phase. The additional symptom in LSD’s favour gave it a positive net number of symptoms, while relegating Heartwater to be discarded soon in the diagnostic process owing to it having a negative net number of symptoms. This results in LSD being given a high confidence, which could indicate the diagnostic strategy should be modified to take this scenario into account. As discussed in Section 4.2.7, a threshold for a number of negative symptoms could be introduced, after consultation with an expert on a recommended number.

4.2.10 The Effect of Immunity on the Diagnostic Process

This test serves to test the functionality of the immunity logic. A number of shared symptoms for both Asiatic Redwater and Gallsickness are confirmed. No symptoms which the CDS could use to differentiate between the two diseases are given. The variables used for this test can be seen in Table 4.11.

The CDS diagnosed Gallsickness. This diagnosis shows that despite the higher likelihood
assigned to Heartwater due to its recent vaccination, the differentiation process shows that
disease is clearly Gall sickness, as the only symptoms not shared between the two diseases
were unique to Gall sickness. The symptoms for Asiatic Redwater were never queried,
despite broad symptoms existing which would cause it to be queried, as a result of the
immunity. An examination of the logs showed that after its immunity was processed, it
was not considered again by the CDS. Should Asiatic Redwater not have been excluded
from contention, a differentiation process would have been needed, and would have been
unsuccessful, as all the confirmed symptoms would have matched both Asiatic Redwater
and Gall sickness.

### 4.3 Evaluation of the UI

The UI was developed to be simple to use and understand, as well as be economical in
the sense of the amount of data required to create it. This was done to allow the CDS to
function even in areas or on mobile phones with only EDGE speeds available for mobile
data.

The system tests were conducted on the official Android 2.2 SDK emulator, using the
Android browser included with the operating system. Some tests were conducted on an
Android 4.2 emulator, with the only differences noted being the subtle styling applied to
buttons and checkboxes, as these take on the default of the operating system on which the

<table>
<thead>
<tr>
<th>Symptoms</th>
<th>Depression</th>
<th>Nervous Signs</th>
<th>Pale to yellow eyes</th>
<th>Pale to yellow gums</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broad Symptoms</td>
<td>Behavioural changes</td>
<td>External physical signs</td>
<td>Loss of appetite</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Immunity</th>
<th>Asiatic Redwater</th>
</tr>
</thead>
<tbody>
<tr>
<td>Herd Infections</td>
<td>None</td>
</tr>
<tr>
<td>Vaccinations</td>
<td>Heartwater (Recent)</td>
</tr>
<tr>
<td>Wet Weather</td>
<td>No</td>
</tr>
<tr>
<td>Month</td>
<td>October</td>
</tr>
<tr>
<td>Cow Age</td>
<td>5 years 0 months</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Diseases for Investigation</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidence (Adjusted Confidence)</td>
<td>Gall sickness: 33 (100)</td>
</tr>
<tr>
<td>Likelihood</td>
<td>Gall sickness: 1.0</td>
</tr>
<tr>
<td>Diagnostic Score</td>
<td>Gall sickness: 100</td>
</tr>
<tr>
<td>Differentiation</td>
<td>Gall sickness over Heartwater</td>
</tr>
<tr>
<td>Diagnosis</td>
<td>Gall sickness (High)</td>
</tr>
</tbody>
</table>

Table 4.11: The effect of immunity on the diagnostic process.
4.3. EVALUATION OF THE UI

browser is running, and the design of the Android UI was modified between the 2.2 and 4.2 releases. All screenshots previously shown in this thesis were taken on the Android 2.2 and Android 4.2 emulators and reflect a point in the diagnostic process of the CDS.

The full CDS was tested in the Android SDK as it was developed using localhost on a normal PC; no server was set up for it on a remote station. This presented no limitations in the testing of the CDS on Android as the SDK includes a fully-functional emulator, allowing for the running of a virtual smartphone, replete with options for RAM, screen resolution and processor.

The interface was, however, further tested on Opera Mini\(^1\) and Nokia’s Xpress Browser\(^2\). Opera Mini remains one of the most popular web browsers on feature phones\(^3\) and Xpress Browser is shipped as the default browser on Nokia’s feature phone and Asha smartphones. Both browsers use server-side data compression to minimise data use by the phones on which they are running. The CDS itself could not be tested on these browsers, as neither Opera’s official mini simulator\(^4\), nor Nokia’s official Asha and feature-phone emulators\(^5\) are able to access a PC’s localhost facility, unlike the Android SDK’s emulator. The UI was, however, tested by modifying the CDS to save the HTML it sent to the client to construct the UI during the normal running of the system to an HTML file. This HTML file was then uploaded onto a free web hosting platform\(^6\) and the web page was viewed using the two browsers. The results of these tests are shown in Figures 4.1 and 4.2. Although the browsers display the content styling slightly differently as a result of the server-side compression, with Xpress Browser in particular showing discrepancies, the functionality of the UI remains intact on these feature-phone browsers\(^7\).

Previous research found a preference for photographs of symptoms to be provided. This feature was, however, not implemented as the knowledge source used for the CDS did not provide photographs and the loading of photographs over EDGE connections, in the quantity required per page refresh based on the design decisions made in Section 3.3.2, could hinder the operation of the CDS. Simple descriptors of systems were favoured over the graphical approach. In the instances where the InfoPaks used technical terms, such

---

\(^1\)http://www.opera.com/mobile/mini
\(^2\)http://www.nokia.com/za-en/support/faq/?action=singleTopic&topic=FA140762
\(^3\)http://www.netmarketshare.com/browser-market-share.aspx?qprid=0&qpcustommd=1&qpcustommb=
\(^4\)http://www.opera.com/developer/opera-mini-simulator
\(^5\)http://developer.nokia.com/Develop/
\(^6\)http://www.000webhost.com/
\(^7\)Nokia Xpress Browser was run during these tests on a Windows Phone 8 smartphone. Despite being run on a smartphone, the capabilities of the browser remain the same as when it is run on a feature phone; the primary marketing angle used by Nokia for the browser on Windows Phone is the reduction in data consumption it provides owing to its server-side processing.
Figure 4.1: The home page of the CDS requesting basic information about a cow on Opera Mini (left) and Nokia Xpress Browser (right).

Figure 4.2: The vaccination query page of the CDS requesting basic information about a cow on Opera Mini (left) and Nokia Xpress Browser (right).
as *lachrymation, emaciation*, or *salivation*, these terms were used in the codebase of the system but were replaced in the queries presented to the user with phrases that were easier to understand, such as “watery, runny eyes”, “the cow is abnormally thin or weak” and “excessive flow of saliva (a large flow of spit from the mouth)”, respectively.

### 4.4 Analysis of the CDS

The CDS performed as expected in the test cases which were presented for evaluation and the diagnoses made by it were generally logical. A possible issue may exist in the rating of a high confidence diagnosis for LSD in the case of the tests conducted in Sections 4.2.7 and 4.2.9. These tests are, however, realistically unlikely in that they presented a large number of symptoms from two dissimilar diseases. Consultation with a veterinary expert may be advised regarding the number of symptoms matched which are contrary to a disease which can be accepted before a diagnosis is rendered to be of an insufficiently high confidence.

While the tests are hindered in that they are not real-life cases and have not been evaluated by a veterinary expert, the tests served to validate the logic used to make diagnoses in this proof-of-concept system. They make sense from the point of view of an understanding of the InfoPak documents provided by the government, which is the level of understanding the intended readers of the documents are likely to possess.

The UI was functional in the mobile browsers it was tested on and, although very basic, served to present the information required of the user in an easy to understand manner with a layout which could fluidly adjust to the screen resolution of the device it was being displayed on.

### 4.5 Summary

This chapter presented an evaluation of the CDS by documenting various test cases designed to test various aspects of the diagnostic process and its decision-making logic and an explanation of decisions made regarding its UI.
Chapter 5

Conclusion

The CDS system which was implemented can accept information from a user about a cow, its health and vaccination history and other sick cows it may have come into contact with. Using information about a broad range of symptoms the animal is displaying, the CDS can determine an initial list of possible diseases which it should query. After querying the user regarding detailed symptoms which the cow is displaying, the CDS utilises this information, and the information about the cow collected previously, to determine a diagnosis. Should the CDS decide upon two diseases which are similar to each other, it will attempt to differentiate between them, failing which it will present them both as potential results. Should the CDS be unable to make a confident diagnosis, it returns no diagnosis.

5.1 Revisiting the Objectives

The objectives in Section 1.1 called for an expert system capable of providing a front-line diagnosis to rural and developing farmers and livestock owners in poor communities, who would otherwise not have access to professional medical care. This expert system was required to be accessible through a low-end mobile phone, in keeping with the income of the targeted users of the system. Lastly, the system should not encroach on the professional territory of veterinarians, but rather enhance the dissemination of information to people who would previously have been unable to access it.

The CDS has demonstrated that it is possible to develop such a system, and has generally succeeded in its goals as a proof of concept. The CDS is accessible on low-end phones
through a mobile browser and can diagnose an illness in cattle from information given to it about a sick cow's symptoms. The use of InfoPaks has meant the CDS facilitates the dissemination of existing information in a better packaged form, rather than seeking to divert the existing work of veterinary professionals to the system. One potential flaw in the CDS is the lack of knowledge about relative symptom importance and real-world testing of it. This could be solved by the involvement of veterinary professionals as consultants on aspects of the system. Such an enhancement would not detract from their work, as the CDS is targeted at individuals who would not be able to make use of their services regardless of the existence of the CDS.

5.2 Future Work on the Expansion of the CDS

Should the CDS be expanded, the following work is suggested:

- Consultation with a veterinary expert regarding the relative importance of symptoms and the extent to which symptoms which have occurred but do not apply to a disease should detract from the CDS's confidence in its potential diagnosis of that disease, or the ability to make such a diagnosis at all.

- Expansion of the number of diseases the CDS is capable of diagnosing.

- The development of a UI allowing the addition of diseases to the CDS by a non-technically skilled person. This UI should not require the manual development of any code or rules by the person adding a disease. The CDS has been developed in such a way that the development of this UI would not involve much effort and should be able to be accomplished within a relatively short space of time. This UI could also allow for the modification of the categories used as broad symptoms relatively easily.

- A case history could be built up by the CDS with feedback presented to it on the accuracy of its previous diagnoses. These cases could be used to assist in making future diagnoses.
References


Perret, S, Carstens, J, Randela, Rendani, & Moyo, Sibusiso. 2000. *Activity Systems And Livelihoods In Eastern Cape Province Rural Areas (Transkei): Household Typologies As Socio-Economic Contributions To A LandCare Project*. Tech. rept.

REFERENCES


Wu, Kangheng, Xiong, Xiaokang, Leung, Bert W, Park, Jihyoun, & Lei, Zhibin. 2012. Event processing of monitoring data of large hi-tech manufacturing equipment: DEBS