BONEDESKTOP: A VISUALIZATION BASED TOOL SUPPORTING THE EVALUATION OF EXTERNAL SKELETAL FIXATION PROPOSALS

by

Eric T. Stiles

(Under the Direction of Khaled Rasheed)

ABSTRACT

Expert System Tools for veterinary students can improve treatments the animals receive and help to train new students. The usability of such a tool is important for acceptance by the target community and for the productivity of users. This thesis involves the development of a visualization-based tool, BoneDesktop, to help with the evaluation of external skeletal fixation proposals. BoneDesktop is also surveyed by Veterinarian Residents, Interns, and Students for evaluation to further enhance its usability in meeting the needs of its target community. BoneDesktop offers its review based on machine learning techniques. In this thesis we compare several variations of two common machine learning techniques, K-Nearest Neighbor and Decision Trees, in their application of veterinary domain data. BoneDesktop then serves as a platform for building future, advanced user friendly features.

INDEX WORDS: human-computer interaction, external bone fixation, decision trees.

BONEDESKTOP: A VISUALIZATION BASED TOOL SUPPORTING THE EVALUATION OF EXTERNAL SKELETAL FIXATION PROPOSALS

by

ERIC T. STILES

B.S.M.E., Auburn University, 1996

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2005

BONEDESKTOP: A VISUALIZATION BASED TOOL SUPPORTING THE EVALUATION

OF EXTERNAL SKELETAL FIXATION PROPOSALS

by

ERIC T. STILES

B.S.M.E., Auburn University, 1996

Major Professor: Khaled Rasheed

Committee: Dennis Aron
Eileen Kraemer

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
May 2005

ACKNOWLEDGEMENTS

I would like to extend my sincerest thanks to Dr. Khaled Rasheed for his valuable guidance, useful suggestions and encouragement, and his time, without which this project would not have been successfully completed. I would like to also thank Dr. Dennis Aron and Dr. Eileen Kraemer for not only being on my committee, but offering far more time and guidance than I'm sure they expected. I would also like to thank the Faculty and Staff of The University of Georgia Computer Science Department for their time and guidance. Finally, I am here because of the constant encouragement and personal support of my parents, Thank you. Poncho and Happy, thank you for making me laugh.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Accidents happen to people and to animals. Bone fractures in animals can be an especially serious problem. Veterinarians must be accurate and precise in their application of a treatment. Animals can only whine or growl when veterinarians assess an injury. Success in applying a solution to a bone fracture is very important to the body's properly healing itself and the long-term health of the animal in general. One of the more popular solutions is known as External Skeletal Fixation (ESF), as seen in Figure 1.1.

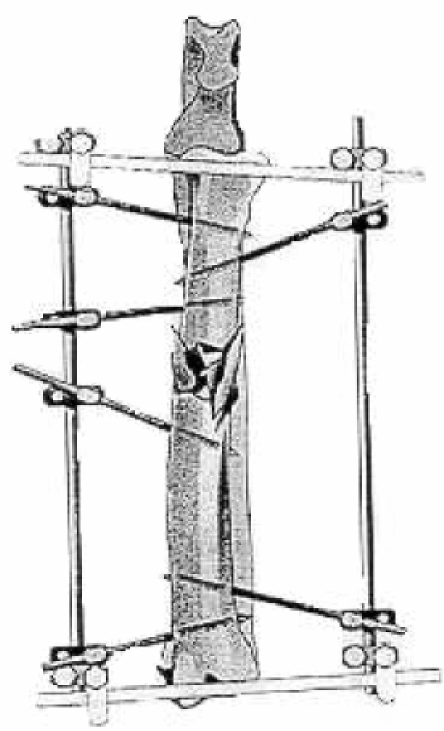

Figure 1.1 : External Skeletal Fixator

The basic external fixator is composed of a series of fixation pins screwed into the bone and attached to an external connecting bar by connecting clamps. The exposed portions of the fixation pins are interconnected using connecting fasteners which attach the fixation pins to the connecting bar(s) [1]. There are three basic types of fixator configurations: Type I, Type II, and Type III. Each type is more sophisticated than the previous one as required by the severity of the injury.

Although fixators can be applied in an almost infinite number of ways, only a few specific fixator configurations will work for any specific type of injury. Then, from those possible solutions only one is considered the best, while a few others might be very good choices. How then can students and new veterinarians learn to filter out bad solutions to find an adequate, if not excellent, fixator proposal? Currently, the only real solution is to analyze injuries as they are brought into the emergency clinic. Textbook illustrations that approximately represent the fracture configuration are then used as a guide. With this method, though, critical case-specific variables such as an the animals mental state are not taken into account. The end goal of the veterinarian is to minimize patient morbidity where morbidity is defined as any departure from complete and normal health [2].

Veterinarians and Researchers at The University of Georgia are working on the development and implementation of a set of attributes to exactly define a case-specific injury and then properly apply an External Skeletal Fixator. Generally known as a Fracture Assessment Score (FAS), these attributes give an in-depth quantitative definition to a qualitative situation. The FAS is made up of 5 categories: Three categories define the fracture, known as the Fracture Case Assessment Score (FCAS), and the remaining two categories define the ESF proposal, known as the Fracture Table Assessment Score (FTAS) [2]. Each attribute is based on a

Table 1.1: An example of the Fracture Assessment Score (FAS).

Biological Fracture Category		Biological Solution Category	
Age	3	Open vs. Closed	4
Metabolic Health	2	Duration of Surgery	6
Trauma History	0	Surgery-Induced Tissue Damage	8
Fracture Type	2	Fixation Implants within the Fracture Zone	5
Fracture Limb Condition	4	Bone Graft	6
Palpation Stability	5	Pin Position	7
Open Fracture	2	**Mechanical Solution Category**	
Other Biological Assessments	3	ESF Device	4
Clinical Fracture Category		Number of Pins and Their Distribution	3
Owner Assessment	3	Pin Location	4
Patient Assessment	4	Type of Pin	4
Mechanical Fracture Category		Size of Fixation Pins	6
Load-Sharing vs. Non-Load Sharing	3	Pin Insertion Technique	6
Patient Weight	4	Load-Sharing	3
Palpation Stability	5	Full-Pin vs. Half-Pins	6
Strength & Function of Other Limbs	4	Number & Separation of Frames	4
Fractured Bone	5	Moment	3
Other Mechanical Assessments	-	Articulations	5
		Supplemental Fixation	3

subjective 1 - 10 point scale where 1 is bad and 10 is excellent. Table 1.1 represents a case example of the FAS in use. The clinical category records how responsive the patient is and how responsible the owner is. Biological attributes such as age, patient history, and various medically defined variables are subjectively scored. The mechanical attributes defining the size of the dog and other medically defined variables are recorded. With the total FAS, veterinarians then apply the scores to a "threshold zone" where the score above this point implies an adequate patient morbidity and increasingly higher scores greatly decrease the patient morbidity. The "threshold zone" is the minimum score where a prescribed ESF is acceptable for the given fracture. The FTAS must complement the FCAS such that the minimum threshold conditions required for fracture healing are satisfied [2]. The resulting score from these attributes is also based on a 10 point scale and is known as the balance score.

1.1 Related Work

The first iteration of this project from Potter [3] and Dale [4] developed a rules-based expert system based on the programming language Prolog. Prolog is a logic programming language enabling the creation of programs as an expression of logic rather than a sequence of instructions leading to a solution. It is a declarative rather than a procedural programming language. A strong feature of this approach is that the domain expert's knowledge is captured by the system. This approach does make it hard to represent the relationships between the existing data within the domain. Duda and Shortliffe [19] distinguish an expert system from a knowledge-based system. They consider the definitions of knowledge-based systems and expert systems vague in providing an exact and unique definition to the point that nearly any Artificial Intelligence (AI) application can be defined as an expert system. They differentiate between the two by defining a knowledge-based system as an application whose success relies on a sufficiently adequate collection of information for a specific domain, rather than a unique procedure providing a solution. An expert system then is a knowledge-based system able to rival in expertise that of a human specialist.

After testing another technique the second iteration of this project started. Dr. Khaled Rasheed, University of Georgia, Computer Science Department, supervised a course project for continuation on this study. Research suggested that the Machine Learning technique known as a Decision Tree was a better approach for predicting a binary decision (i.e., whether an ESF was good or bad) [6]. Suo [6] worked towards the new application in testing a Decision Tree against other possibilities. The positive results regarding a Decision Tree helped to continue providing validity to Dr. Aron's research.

1.2 Objectives of this Study

The GUI interface prior to this research did not help with the "bottleneck" of knowledge acquisition. It was trivial at best. The client focus was designed for an expert computer user absolutely familiar with the program and the machine learning techniques behind it. Certainly it was not a production-ready application for the typical veterinarian. The overall goal then of this thesis and research is the quality development of a visualization-based tool, BoneDesktop, to help with the evaluation of external skeletal fixation proposals and input of new domain data for testing purposes. Only a beta version is required as a full-blown production release is beyond the scope of this Master's Thesis. The development process for this thesis involved the following steps:

1. Design flow process of the program including the backend controller logic and front end visualization tool.
2. Survey existing tool and review for modification as necessary.
3. Review two Machine Learning Techniques (Decision Tree and K-Nearest Neighbor), then compare them based on specific qualities of the supplied veterinary based data.

The layout for the rest of the thesis is as follows. Chapter 2 reviews the requirements analysis. Chapter 3 discusses existing code, database, design patterns and architecture implementation, the user interface, and its development including internationalization and configuration files needed. Chapter 4 then runs through the workflow and a case scenario. Chapter 5 discusses the user reaction and design changes needed. Chapter 6 relates research into comparing the implemented Decision Tree with the K-Nearest Neighbor as an alternative approach. Finally, chapter 7 offers ideas for future work and the final conclusions of this thesis.

CHAPTER 2

REQUIREMENTS ANALYSIS

The history surrounding the work with this project represents work in several different areas, from an early expert system, to an analysis of various learning methods, including the very development of a Finite Element Analysis (FEA) tool. These separate projects were successful within their limited scope of design and use. Through these and other projects Dr. Dennis Aron, the current project supervisor and domain expert for this thesis, had results but did not yet have a fully running usable application. The resounding goal then is the development of a tool that not only meets existing criteria but was fully ready for future review and development beyond the completion of this thesis.

The focus in the development of BoneDesktop was a user-centered approach [13]. The early focus is on all users and their tasks. By observing and discussing with the intended clients in their environment, we can better understand their mental model of the system and tasks.. Early development should also focus on empirical measurement such as reaction and interest in various printed simulations and throw away prototypes. Then through an iterative process of “design, test, measure, and redesign” the project will evolve into a successful application. Initial discussions with Dr. Aron revealed his desire to have a tool that allowed for the input of all necessary scoring attributes and the resulting decision of whether the given fixator proposal was excellent, good, or poor. His experience with computers is minimal and he wanted a system for that type of user. He also expressed interest in viewing medical images as well as other specific goals. His review of initial paper mock-ups yielded several designs to start the iterative process with. Beyond that, full autonomy was given to best implement the proposal. Then, reviewing early throw-away prototypes with Dr Eileen Kraemer, University of Georgia, Computer Science Department, helped produce a more user-friendly product. The following guidelines helped to minimize scope creep and still allow for a fully functioning visualization tool:

1. There are three types of users for this program: The client user, administrator, and programmer. Each role is specific and requires equal consideration in its importance, although most users will fall under the first category.
2. The framework of the application needed to be extensible and properly documented to allow for easy updates over time as the application is developed and new users come into the project while other programmers leave. Proper documentation also helps to increase the project lifecycle.
3. The early prototype is based on a minimal set of domain data. Future work absolutely requires more data to improve both the learning techniques that are currently in place and a student's ability to research injuries they haven't yet come across in an emergency clinic. Information needs to follow the basic CRUD guidelines (Create, Retrieve, Update, Delete).

2.1 User Considerations

Identifying the needs and establishing requirements is fundamental to successful interaction design. The initial surveying and research for this study revealed that the majority of the clients for this application were, at a minimum, comfortable with using computers, but they were only familiar with the Microsoft based operating system. From earlier statements, BoneDesktop must be comfortable for users with minimal experience as well as users comfortable with computers. A standard set of common mnemonics [14] is required. Although this application does not offer files that are opened, reviewed, and saved, it does require the CRUD operations for information located in a data store. The users of this program can vary from administrative assistants, to veterinary students, interns, residents, and doctors. The primary focus of BoneDesktop is for medically trained users. The terminology must be focused enough that its application is seamless within a veterinarian's work environment. The way information is displayed can influence how easy or difficult it is to assess the content within that information. With the large amount of data for each specific injury it will be important to group the information appropriately to ease the student's ability to comprehend the current set of data.

2.2 Administrator Considerations

The Administrator understands how the program runs and what is needed to set up an application in their working environment. In this case the administrator is considered to also be knowledgeable with veterinary terminology. With the implementation of internationalization [15], an administrator can use locale specific terminology. The administrator should have the ability to alter the text to better give the client user a higher comfort level with the software without needing to rebuild the application. Internationalization will be implemented. Also, specific to this program, the administrator will have the ability to alter scoring attributes by removing old ones, adding new ones, or reordering existing attributes.

2.3 Programmer Considerations

The programmer should have the ability to easily write extensible plug-ins for the software based on the needs at that time. The easiest way to help future programmers work with software is to have fairly high coupling between the various portions of the code. By focusing on development of the Application Programmer Interface (API) as a separate entity from the User Interface Implementation the important methods can be fully developed without concern for its use in the user interface. Then opening the methods up with proper documentation will help with proper implementation. Also, BoneDesktop forces users to extend base classes in the user interface hierarchy to makes sure programmers write code properly implementing specific interfaces.

CHAPTER 3

APPLICATION IMPLEMENTATION

In order to complete this study and finish with a quality deliverable, several well-known and successfully tested projects, design patterns, and approaches were used. The Open-Source Movement is extensively used in getting tested, baseline code to work from. Those projects include Ant [17], JScroll [12], XML parser (Xerces [18]), and MySQL (although MySQL was used as the data store, any data store is easily implemented). Various design patterns including Factory pattern, Observer Pattern, Singleton Pattern, Iterator Pattern, and the Command Pattern were implemented. Few projects proceed totally as planned, thus, there must be a plan of action; the development of this visualization tool follows the learn-as-you-go approach to software development, more formally known as iterative and incremental software development [7].

3.1 Open Source Initiative

The Open Source Initiative (OSI) [8] has provided the programming community with an environment to comfortably develop various applications. The goal behind OSI is to allow programmers the opportunity to read, redistribute, and modify the source code for a piece of software. The software evolves as people improve it, adapt it, and fix bugs. From the OSI movement products such as Ant, JScroll, MySQL, and Xerces developed into quality solutions.

Ant is the de-facto industry standard build tool for Java applications. Using build files based in XML Ant, a powerful tool, allows for cross-platform development and extensibility. In this application Ant is able to build a complete application that allows an administrator to quickly and easily drop in a running application with minimal effort.

The JScroll [12] framework implementation is a seamless enhancement to existing Java Multiple Document Interface (MDI) applications. JScroll extends already existing Java Swing component classes to provide a virtual desktop environment replete with dynamic scroll bars, dynamic menus, and dynamic button shortcuts. Figure 3.1 is a representation of BoneDesktop's implementation of JScroll.

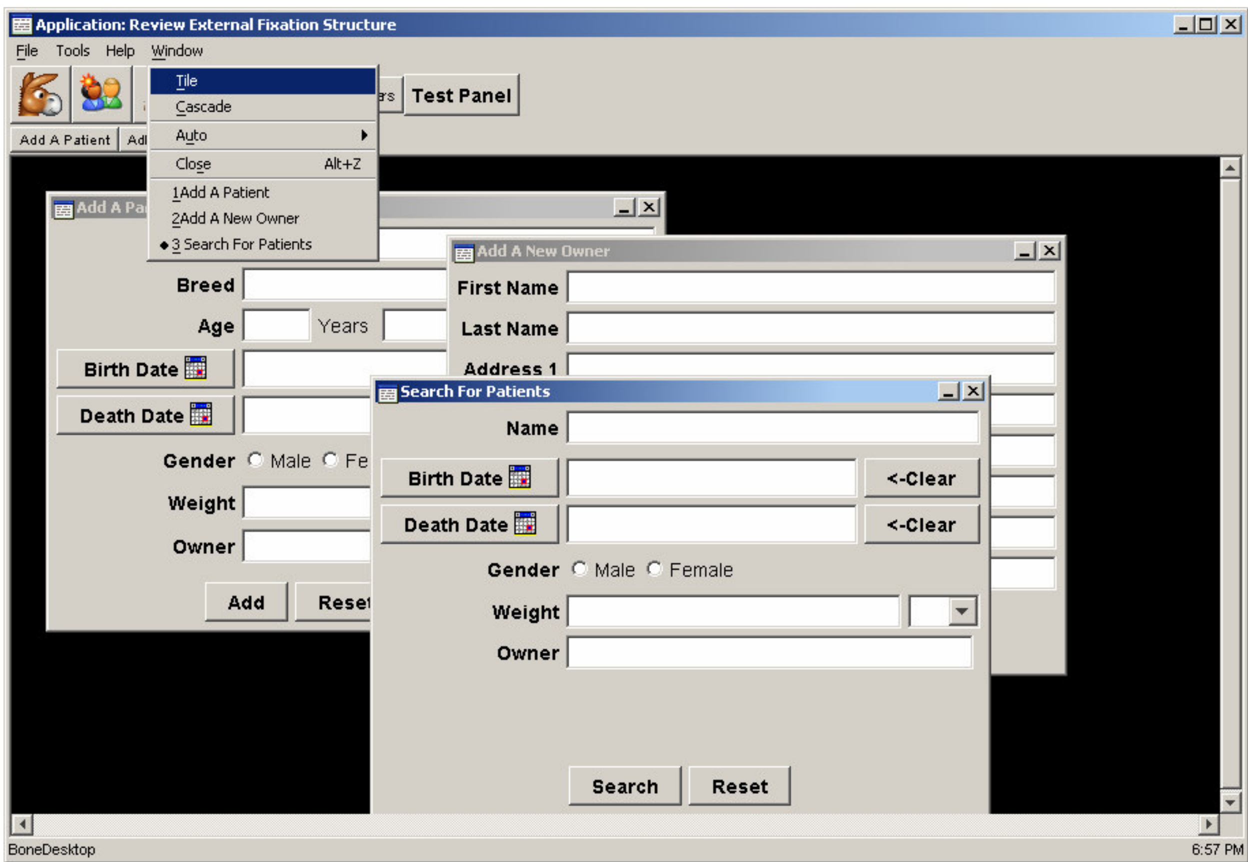

Figure 3.1: BoneDesktop implementing JScroll

BoneDesktop is not dependent on any specific database. For testing purposes and initial releases to Dr. Aron, the MySQL [9] database was chosen for its stability, ease of implementation, and cost (free). BoneDesktop uses a properties file to define the required connection properties. Those properties are the specific connection bridge for the given data store, a URL for the database, a username, and requisite password. If so inclined, an implementation storing all text in a flat file structure is possible. A flat file implementation would require programming a new database middle layer class implementing the methods from the database connector interface. Then using that code in place of the standard database middle layer would allow the use of a file structure.

Extensible Markup Language (XML) [10], developed as a common text format for structuring data, content allows for a common platform to quickly process information between different applications. BoneDesktop uses XML to format the scoring attributes system. Figure 3.2 is a partial example of BoneDesktop's use of XML in describing the scoring system.

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<configuration>
<injury>
    <clinical>
        <attribute title="Patient" order="0">
            <value id="1" view="Active and Uncontrollable" />
            <value id="2" view="Active and Uncontrollable" />
            <value id="3" view="Active and Uncontrollable" />
            <value id="4" view="Active, but sensible" />
            <value id="5" view="Active, but sensible" />
            <value id="6" view="Active, but sensible" />
            <value id="7" view="Inactive, cooperative" />
            <value id="8" view="Inactive, cooperative" />
            <value id="9" view="Inactive, cooperative" />
            <value id="10" view="Inactive, cooperative" />
        </attribute>
        <attribute title="Client" order="1">
            <value id="1" view="Nonreliable" />
```

Figure 3.2: Example of the XML Scoring Configuration File

3.2 Implemented Design Patterns

Design Patterns is a documented best practice successfully applied in multiple environments to solve a problem that recurs under a specific set of situations [11]. BoneDesktop incorporates the following patterns:

- Command Pattern: Object-oriented design consists of interacting objects; each offering focused functionality. When needed, the application calls different objects as required.
- Factory pattern: Subclasses in a class hierarchy generally inherit the methods implemented by the parent class. Subclasses can override the parent class implementation to offer a different type of functionality for the same method. When an encapsulating object knows the exact functionality needed, that class is directly instantiated from within the class hierarchy. When an application does not know which specific subclass it needs only that it is the within the parent hierarchy the factory pattern returns the proper object.
- Mediator Pattern: Large scale object oriented designs consist of a collection of objects interacting with each other to provide a specific service. With a large number of objects the communication between them can be very complex affecting the maintenance of the application. The Mediator pattern is a traffic cop directing the communication between objects reducing the scope of the objects and creating a higher coupling.
- Observer Pattern: The Observer allows for a consistent communication model between a set of dependent objects and any object they are dependent on. Dependent objects then have their state synchronized with the object that they are dependent on. Dependent objects are known as the observers and the object that they are dependent on is the subject.

- Singleton Pattern: The Singleton pattern is implemented to control the instantiation and use of a single specific class. When only one class is needed to sufficiently complete the task there is no need to increase the program footprint and CPU process time with the instantiation of many classes.

3.3 Architecture

The focus of this research is a visualization tool. BoneDesktop can only come to fruition if there is a proper framework to work from. A proper architecture implementing the aforementioned design patterns will greatly reduce the amount of developmental redundancy. A by-product is the development of a common set of GUI panels and their implemented functionality that users will quickly come to understand. Figure 3.3 represents a simple layout of how the architecture was developed and various points to hook into when using the program.

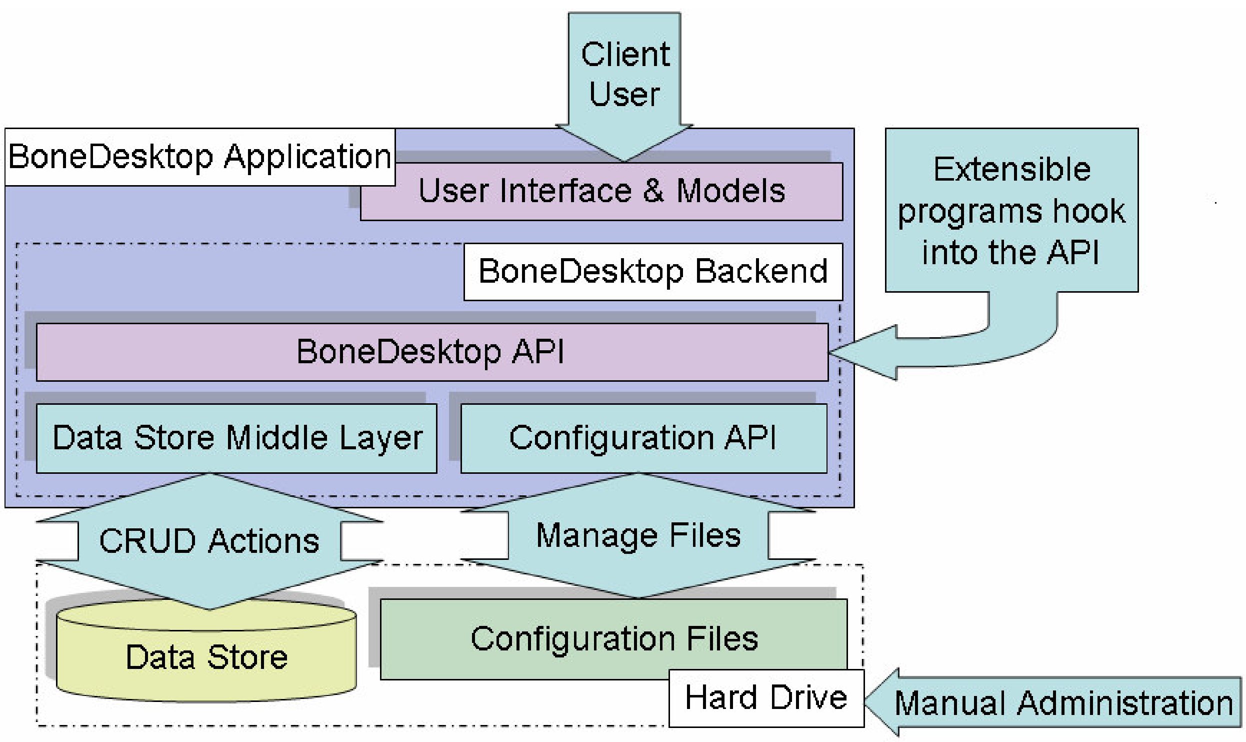

Figure 3.3: BoneDesktop Architecture

Figure 3.3 does not explicitly demonstrate the application of the various designs as that is beyond the scope of this thesis.

The Data Store contains all content related to the application. That information includes data about a pet, its owner, any injuries, and their proposed solution(s). MySQL was used as the database for development and testing purposes, but an administrator can quickly change to another database of choice. As part of the overall application requested for installation, a SQL file is given as the BoneDesktop requires specific tables be set up before it will run successfully. Appendix A contains the SQL table data for BoneDesktop.

Figure 3.4 is the layout of the for the database schema. It is not a complex schema. All primary keys are strictly made of unique integer id autoincrementing when a new tuple is added. The schema is fairly basic where the attribute names are self-explanatory. The BAC#, MAC#, and CAC# are the Biological, Mechanical, and Clinical attributes and a number identifying which one they are referring too.

There are several different configuration files BoneDesktop uses. An XML file contains the attribute scoring information and is loaded once on startup. A file located in the user's home directory stores session information such as location and size information about the BoneDesktop window. Internationalization [15], the process of designing an application so that it can be adapted to various languages and regions without engineering changes, is used. Within the supplied jar is an internationalization properties file containing the value-label pair for each word or statement in BoneDesktop. The internationalization file is loaded at startup. The virtual machine that BoneDesktop runs on top of will automatically choose the specific locale for the language on the computer if it exists; otherwise a default language file is loaded. BoneDesktop only uses a Decision Tree. It does not learn from any new supplied data.

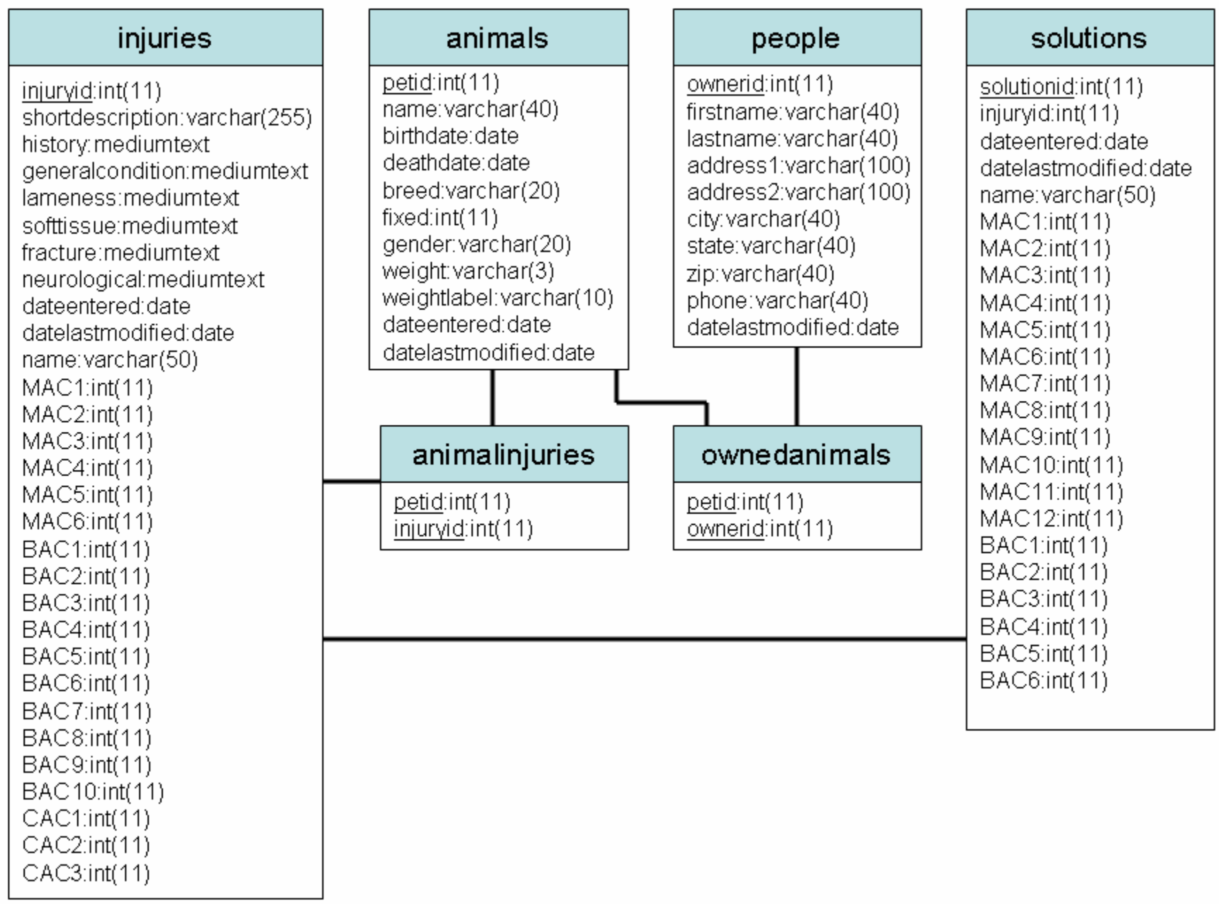

Figure 3.4: BoneDesktop database schema

The program See5 [16] is used for the development of the Decision Tree. BoneDesktop then parses a default output file or any other user supplied See5 output file. Figure 3.5 is a partial example of a See5 output file before it is parsed.

The architecture of the program is focused on compartmentalizing sections of the overall application. There are three main sections of the code. The view as represented by Figure 3.3 separates the stored content, the logic, and the view. That separation allows for a program that is loosely coupled, reducing unnecessary redundancy and improving a programmer's ability at re-factoring. The design of this API includes hooks that the GUI interfaces use to communicate with the BoneDesktop application. Future programmers will use these hooks to design their own internal frame to run in the application MDI. One such example is the future development of a medical imaging module for x-rays.

```
See5 [Release 1.20a]    Tue Feb 22 23:54:55 2005
--------------------

Read 60 cases (4 attributes) from dog_orig_ave.data

Decision tree:

MechanicalTreatmentScoringAverage > 5.91:
:...BiologicalTreatmentScoringAverage > 5.17: 10 (14/5)
:   BiologicalTreatmentScoringAverage <= 5.17:
:   :...BiologicalInjuryScoringAverage <= 3.4: 8 (2)
:       BiologicalInjuryScoringAverage > 3.4: 6 (8/3)
MechanicalTreatmentScoringAverage <= 5.91:
:...MechanicalTreatmentScoringAverage > 4.3:
    :...MechanicalClinicalInjuryScoringAverage <= 5.1: 5 (7/3)
    :   MechanicalClinicalInjuryScoringAverage > 5.1:
    :   :...BiologicalInjuryScoringAverage <= 4.25: 4 (2)
    :       BiologicalInjuryScoringAverage > 4.25: 3 (5)
    MechanicalTreatmentScoringAverage <= 4.3:
    :...MechanicalClinicalInjuryScoringAverage <= 3.9: 2 (3/2)
        MechanicalClinicalInjuryScoringAverage > 3.9:
        :...MechanicalTreatmentScoringAverage > 3.82: 2 (3)
            MechanicalTreatmentScoringAverage <= 3.82:
            :...MechanicalTreatmentScoringAverage <= 3: 1 (8/1)
                MechanicalTreatmentScoringAverage > 3:
                :...BiologicalTreatmentScoringAverage <= 3: 1 (4/1)
                    BiologicalTreatmentScoringAverage > 3: 3 (4)
```

Figure 3.5: See5 example output file

3.4 User Interface Implementation

Usability is generally regarded as ensuring that interactive products are easy to learn, effective to use, and enjoyable from the user's perspective. It involves optimizing the interactions people have with interactive products to enable them to carry out their activities at work, school, and in their everyday life. Key information needs to be salient at all times. Usability is broken down into the following goals: effectiveness, efficiency, safety, utility, learnability, and memorability [13]. BoneDesktop strives to meet those goals.

BoneDesktop is designed to encapsulate the FAS scoring system. Other information must be recorded to help develop a workable application with proper flow. Owner information is important for tracking and consistency among owner's pets and injuries. Qualitative information pertaining to the injury is also needed. While it is nice to define an injury in terms of a point system, there is a necessity to record a written medical history. With the addition of other necessary data, a large amount of information is preserved for each case reviewed. By breaking the data into similar chunks of data the user is easily able to enter related key information. BoneDesktop implements a Multiple Document Interface (MDI) to allow users to keep those sections of information up while entering new information (Figure 3.1 represents several panels open at once). The advantage is that users only need to focus on the predominant panel while having panels either in the background or minimized and visibly out of the way.

When trying to offer a full-fledged application, context menus, menus that appear when right clicking over a specific part of an application, need serious consideration. There wasn't much functionality implemented outside of the flow of the program, but the groundwork was laid for future development using context menus. With users needing to type a substantial amount of information, there is a necessity to duplicate information in various portions. Thus, the common cut, copy, paste, and delete commands were implemented for the medical notes section of an injury as context menu choices (they are also implemented with mnemonics).

The design feature requiring the most time and research was the scoring system for the injury and treatment. It is the key feature of BoneDesktop and is not a standard feature of GUI design. The description of the score is important for the user to know what value to choose. Thus, the user is more interested in a definition to choose rather than a point value. Initially, paper templates were developed to have Dr. Aron review and give his thoughts on the usability.

Figure 3.6 is the first of several throw-away prototype designs. The initial combo box design did not offer the usability features BoneDesktop needed. The design does not allow users to see all score definitions at once; also, there is more intensive use of the mouse to maneuver through the design.

The idea then moved to Figure 3.7, a slider design that improved a user's ability at scoring an injury with mouse movements. It did not, however, allow users the important feature of seeing all definitions for each point in each scoring attribute. During the development of the third iteration for the scoring GUI screen, Dr. Kraemer suggested that users should see the point definitions all the time, rather than just seeing the definition for the chosen point. Figure 4.6 is the current scoring design iteration. The client can quickly make changes with the movement of the mouse, and is able to see the other choices prior to changing the specific attribute. When a user does makes a change the slider tab box turns yellow notifying the user of changes.

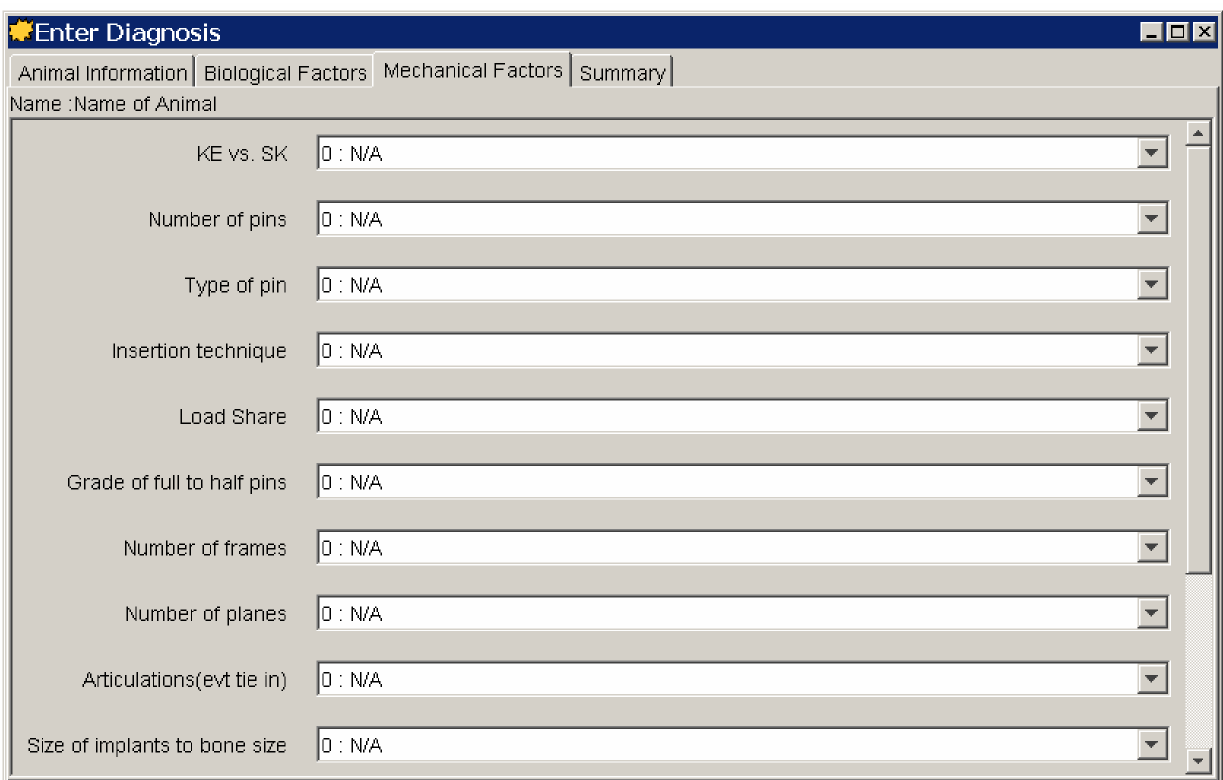

Figure 3.6: Initial Throw-away Prototype Design For Scoring System.

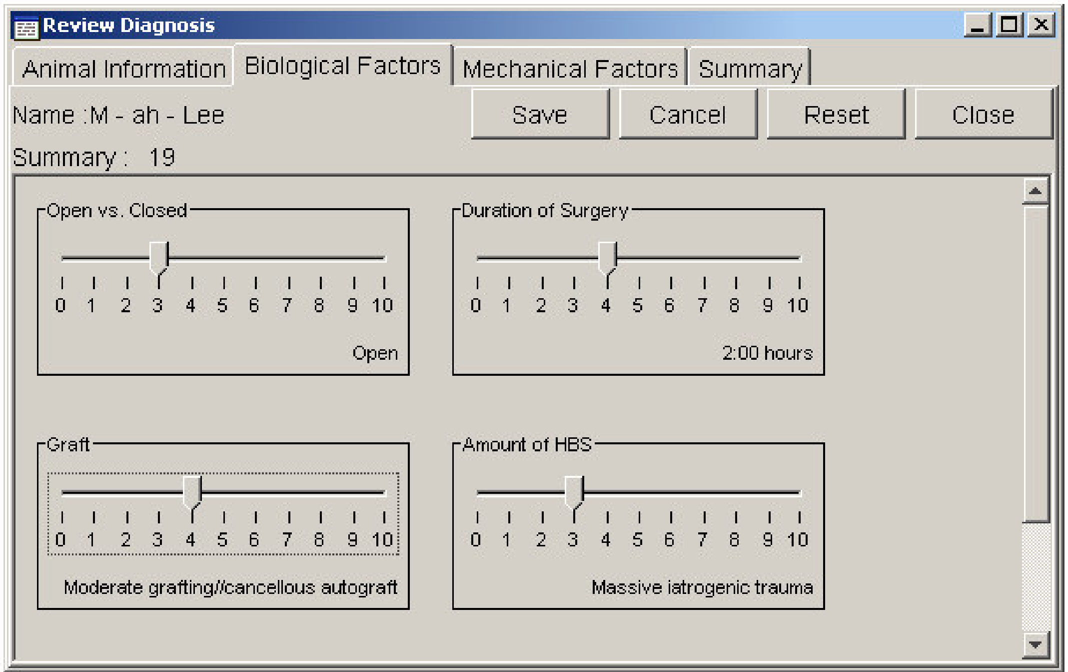

Figure 3.7: Second Iteration Of GUI Scoring Panel. This interface allows better mouse usage, but does not let users see the definitions of the attribute points not chosen.

CHAPTER 4

WORKFLOW AND A CASE SCENARIO

Veterinary students needed a visualization tool to learn how to apply the best ESF. There are two ways that BoneDesktop helps the student accomplish those tasks. The first scenario is through its application as an actual medical database recording emergency clinic patient information. The second scenario allows the student to create test injuries and their complementary proposals to understand how their choices fit with real world data.

A specific case example will best illustrate the workflow. Nero, a Labrador retriever, is brought into the emergency clinic. Initially the administrative assistant will enter information as reviewed in Table 4.1. As the assistant starts up BoneDesktop, Figure 4.1 represents the initial view of BoneDesktop with a description of what the buttons mean.

Table 4.1: Case Example General Patient Information

Patient Information	
Breed	Labrador Retriever
Age	4 years
Gender	male
Weight	60 pounds
Condition	Good
Character of Patient	Active, very nice and adequately trained.
Other	-
History	HBC 5 hours ago by car, hit on hind part of dog, dog "flew away" after crash on a soft grass field, car - speed of 55 mph

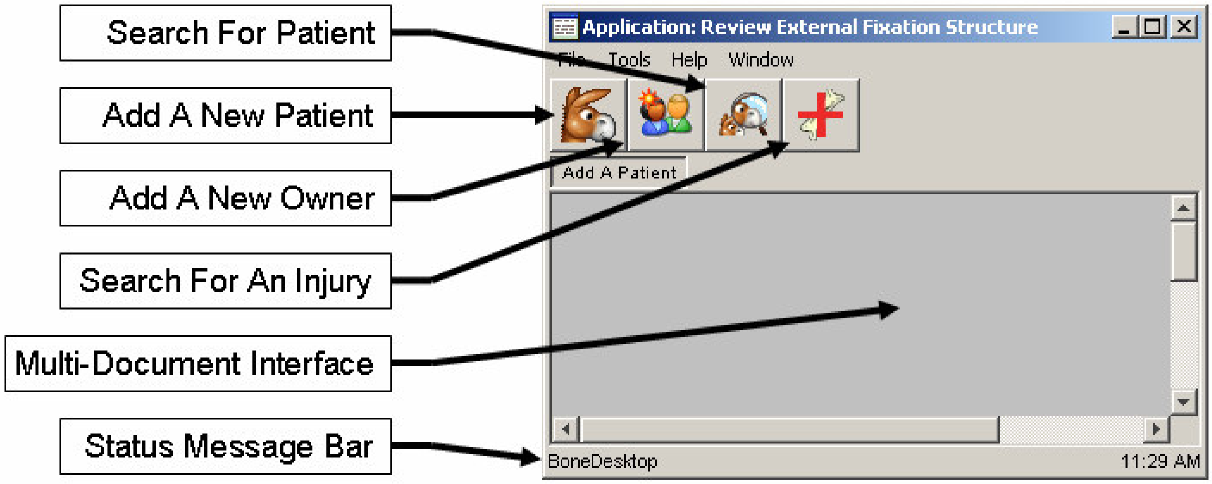

Figure 4.1: BoneDesktop GUI On Startup

The toolbar icons represent the functionality available while the application is running. When adding a new patient clicking the proper toolbar button opens the panel represented in Figure 4.2. In order to compensate for the common lack of consistent information pertaining to an animal's birth the age can be entered, upon which a relative birth date is chosen based on that age, or a birth date is chosen and the age automatically entered into the proper age fields

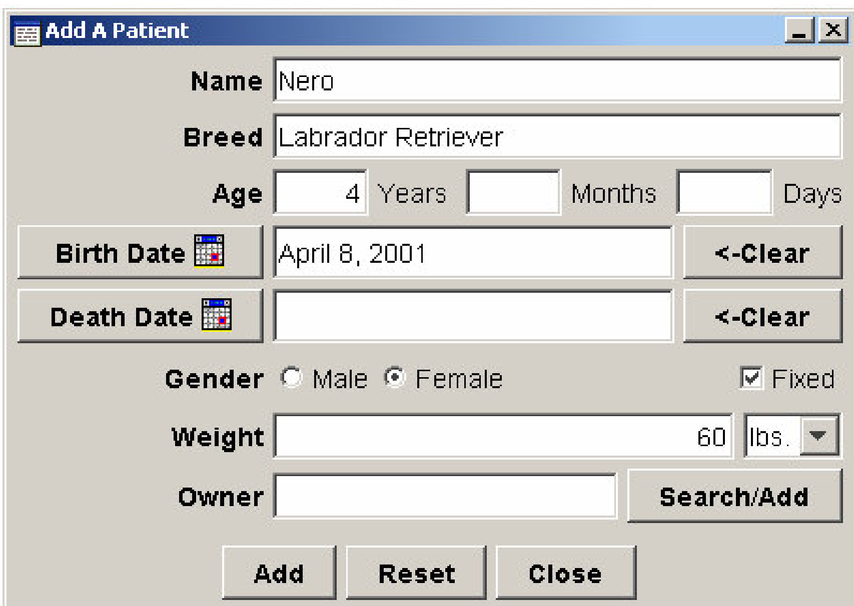

Figure 4.2: 'Add A Patient' panel filled out.

It is not a requirement, but by typing the owner name in the owner text field then clicking the 'Search/Add' button a list of existing owners appears or if the owner is new a panel to enter new owner information appears.

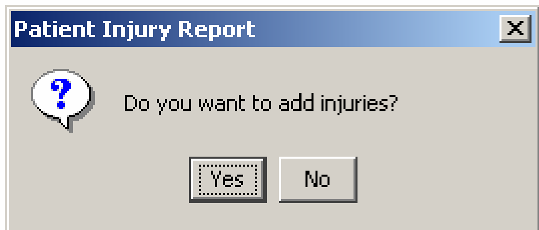

Figure 4.3: Add injuries directly after adding a new patient

After the patient information is entered there is an option to enter new injuries directly. In this scenario the administrative would hit no (Figure 4.3). The veterinary student then comes in and brings the animal into the back for review. Table 4.2 contains the notes taken from the physical examination that occurred.

Table 4.2: Notes From Nero's Examination

Physical Examination	
General Condition/Pain	Good condition, but very painful
Lameness	Non-weightbearing lameness on **left**
Soft Tissue	Severe swelling, extended wounds on
Fracture	**Midshaft comminuted open (grade II)**
Neurological Examination	o.k.
Other	-
Owner	Familiy owns the dog. Mother always at home and dog is really well taught. Family very willing and able to take care of dog, no holidays planned for the next six months. Very reliable owner. Dog in a good standard health, adequately trained. Dog is active but cooperative.

Now the student is ready to enter the information pertaining to the injury into BoneDesktop and work toward a proper solution. Clicking the 'Search Patient' icon button as shown in Figure 4.1 makes a search panel (Figure 4.4) appear.

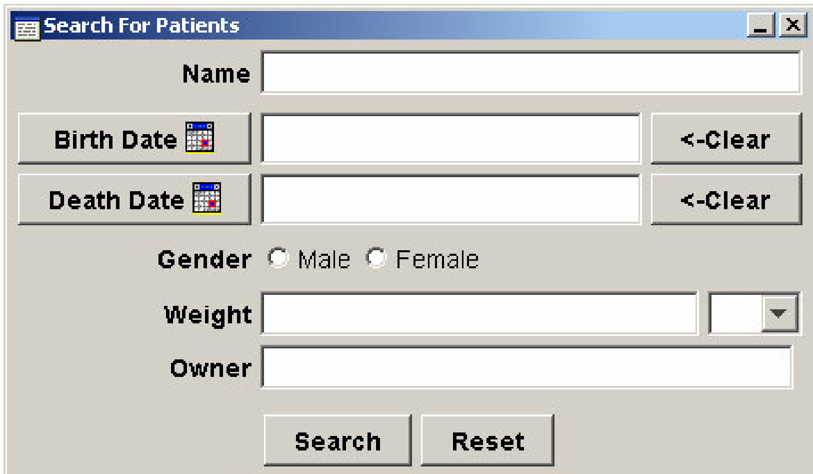

Figure 4.4: Search Panel for patients. Notice the similarities with the panel in Fig 4.3. An implementation of the factory pattern allows for a common user interface panel completing two very distinct tasks.

After typing in the patient's name (Nero) and clicking 'Search', then retrieving the patient from the search list a panel of general patient information opens (Figure 4.5).

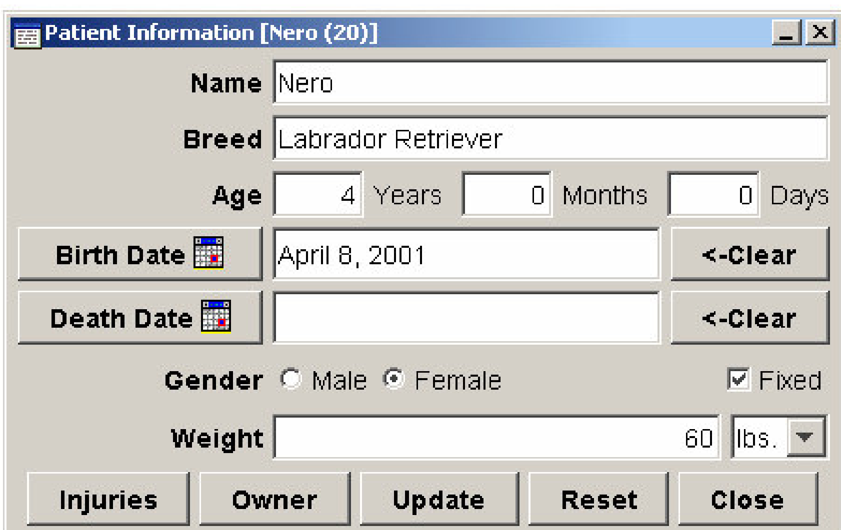

Figure 4.5: Panel of patient information leading into patient injuries.

When the 'Injuries' button is clicked a panel listing the injuries is shown. Our case example is a new patient, thus there are no listings in Figure 4.6. If any injury existed then user's can either highlight the injury and click the 'Display' button or double-click the injury listed.

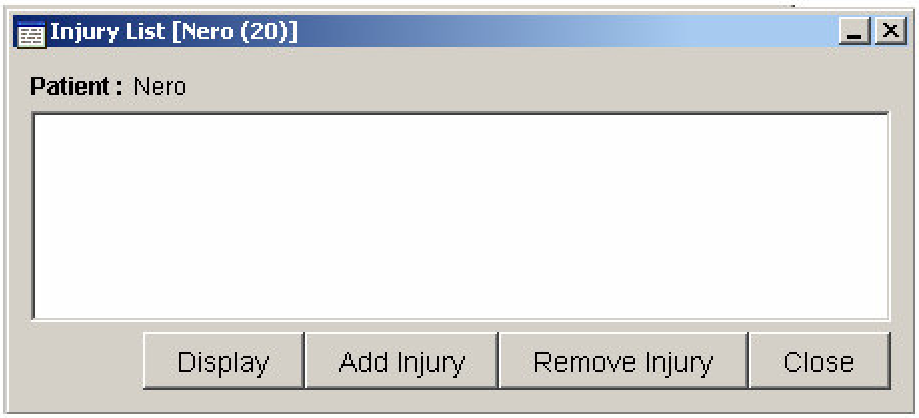

Figure 4.6: Search List of Injuries. The main entry point for all injuries. In this case there are no injuries for Nero.

In this case, though, the student needs to click the 'Add Injury' button. From this point the parallel of students learning how to treat new injuries is the same process as entering a new injury from the medical clinic. Users will click on the 'Search Injuries' to find the desired injury. Figure 4.7 is the panel used to enter the medical information. The title bar of the internal frame lists the fact that this is a 'New Injury'. Also, the internal id of the injury is listed. BoneDesktop is still a beta version and the internal id is viewable in order to allow users the ability to backtrack any problems that occur. The injury information is broken up into 5 distinct categories: basic animal information, injury information, medical information gleaned from the physical exam, then the biological attributes and mechanical attributes are defined using a slider similar to that viewed in Figure 4.7. Although there are slider values for the age and size of the animal they are locked into place. When the sliders are adjusted they turn yellow to let the student know

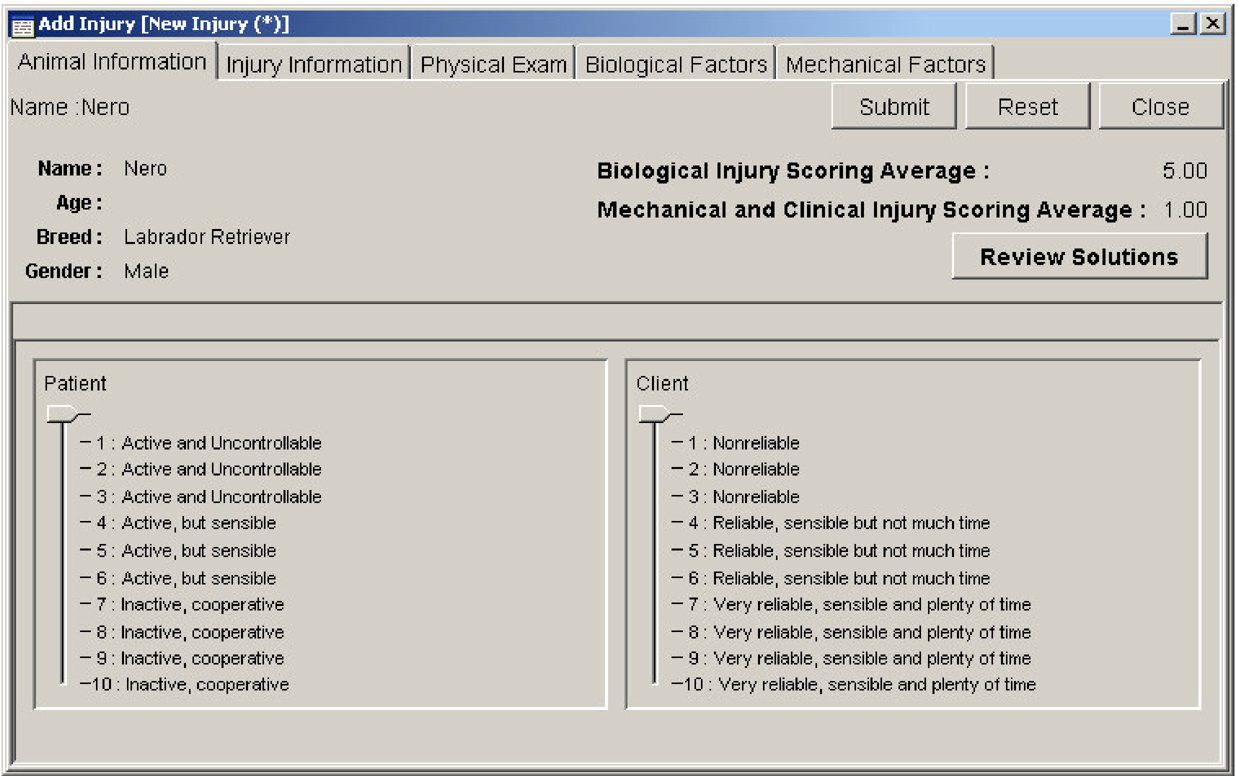

Figure 4.7: Enter injury panel. Tabbed panes continue to separate data in common grouping of information.

they are modified. As well, the title bar on the internal frame is changed to include the notice that the injury is 'modified'. The panel lists a scoring average for the biological attributes and the combination of the mechanical and clinical scoring attributes. Chapter 6 will discuss in depth how the values are used within BoneDesktop.

The 'Review Solutions' button searches for all the solutions for the given injury. In this case there are no solutions. A search results panel similar to Figure. 4.6 is created (listing 0 current solutions). Clicking for a new solution follows with Figure 4.8. The tabbed panes again separate data into common areas of information. The 'Solution Information' pane lists the injury information as well as the attribute scoring averages (Chapter 6 will review the implementation of the averages).

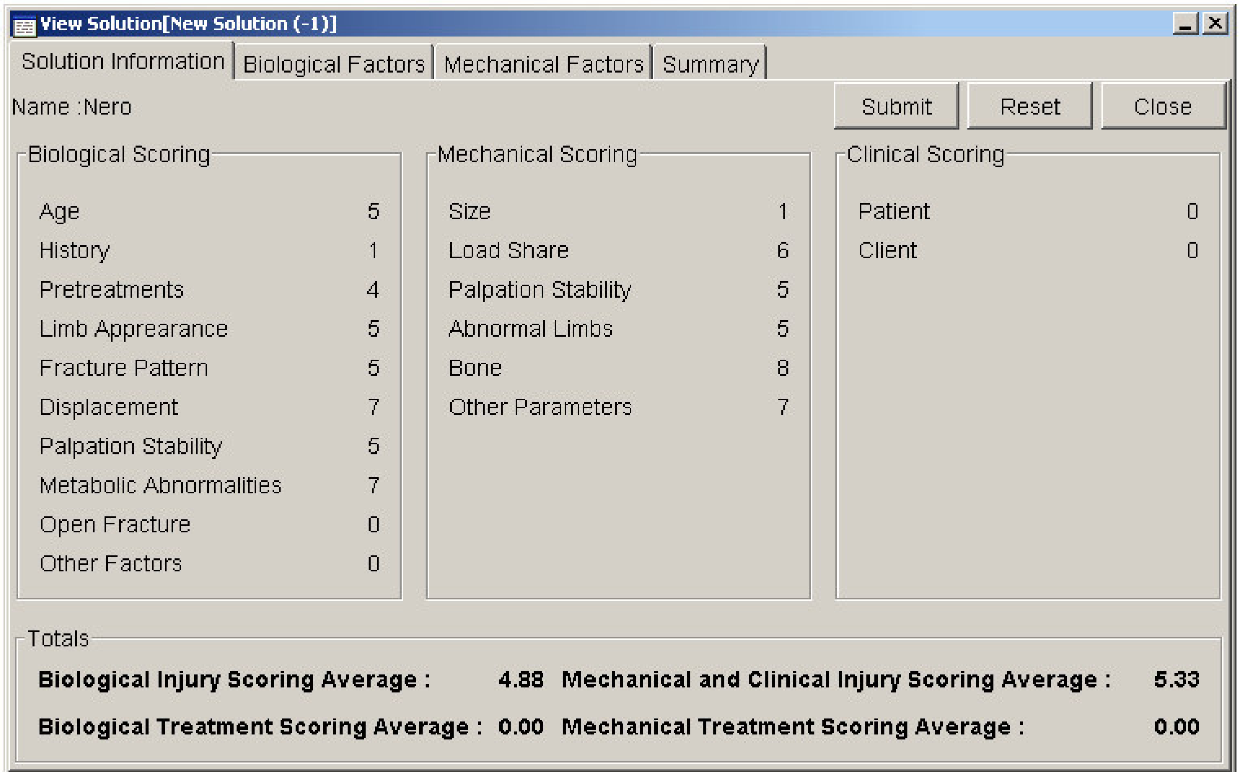

Figure 4.8: Enter Solution Panel. Again tabbed panes help to distinguish common breaks in information. Another factory pattern implementation helps with common GUI components.

The ESF proposed for Nero had the attributes in Table 4.3. It is important to understand that at this point it is not known whether the proposed solution is a good one. Aron [2] defines exactly what the proposal attributes mean. After the attribute values are entered in the solution panel the student will tab over to the 'Summary' panel.

Table 4.3 : External Fixator Proposal Parameters

External Fixator Treatment Parameters			
Biological		**Mechanical**	
1. Open vs. Closed	10	1. KE vs. SK	8
2. Duration of surgery	4	2. Number of Pins	10
3. Tissue observation	2	3. Type of Pin	10
4. Amount of implants	-	4. Insertion technique	10
5. Graft	-	5. Load Share	1
6. Amount of HBS	5	6. Grade of full to half pins	7
		7. Number of frames	10
		8. Number of planes	6
		9. Articulations (evt Tie in)	7
		10. Size of implants to bone size	9
		11. Supplemental fixation	-
		12. Pin Location	10

The 'Summary' panel is the culmination of entering the FAS attributes. The student can choose the rule set they desire or keep the default rule set (Chapter 6 covers in depth what purpose of the rule set is). Once chosen the rules that this decision is based on are shown. Understanding the rules is not necessary for the Decision Tree to provide a meaningful answer. Reviewing the rules can help to make slight variations in the decision to improve a proposal. Finally, the Decision Tree is run. From the example we are working with the proposed fixator is the best choice possible with a value of 10. The color legend quickly lets a user visualize their decision.

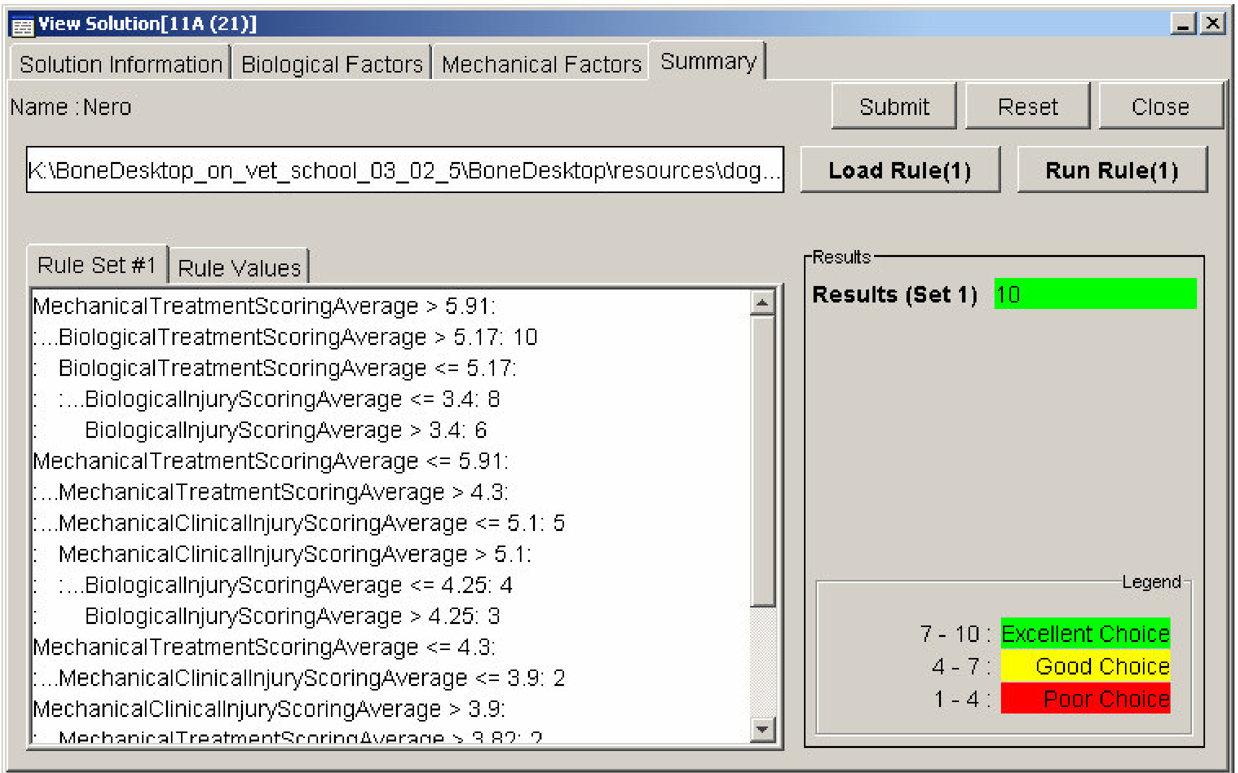

Figure 4.9: Summary View From 'Enter Solution Panel'.

CHAPTER 5

USER REVIEW AND IMPLICATIONS

Designers assume that if they and their colleagues can use the software and find it attractive, others will too. Furthermore, they prefer to avoid doing evaluation because it adds development time and costs money [13]. The basic premise of the user-centered approach that we have ascribed to in the development of BoneDesktop is that users' needs are taken into account throughout the design and development. The design proceeds in an iterative cycle. Continuous interaction with veterinary students is not realistic due to their busy and emergency-oriented schedule. Dr. Aron, the domain expert, served as a quality substitute during the initial development iteration cycles.

The Nielsen Norman Group (http://www.nngroup.com), a usability consultancy company, point out that "User experience" encompasses all aspects of the end-user's interaction. The first requirement for an exemplary user experience is to meet the exact needs of the customer, without fuss or bother. Next, is simplicity and relevance that produce products that are a joy to own, a joy to use [13]." With a running beta version of BoneDesktop, the research did need to get quality feedback from the veterinary students who would use this product on a daily basis. A survey (Appendix B) was provided to 8 students to see if BoneDesktop did, in fact, provide simple and relevant functionality without deterring the client's desire to use it. The survey was given to allow students an adequate process for responding to three different scenarios that tested all of the required functionality and most of the other provided functionality.

This first survey provided a great amount of initial and useful insight into who the users were and what they thought of BoneDesktop. Of the 8 users, 2 were Residents, 2 were Interns, and 4 were Students, a good sampling for the clinicians in the emergency clinic. Nearly 90% of the users were generally comfortable, or did not feel uncomfortable with computers. This is important as a comfortable user is more adept at figuring out how to solve the scenarios. None of the users had ever reviewed software before, which is positive as they don't have any previous experience to internally compare the current experience and can judge the software on its own merits.

The students were comfortable with the design with only 13% finding any of the screens confusing and 88% of the students finding the overall program easy to use. The biggest issue that did start to present itself was the high number of internal documents that could be opened within the MDI. With one injury and 1 solution open, a user could conceivably have 5 screens open. While users can minimize some screens, it was apparent in watching them that the students did not utilize that feature and later commented that at times the interface was confusing due to the large number of open screens. The next redesign needs to incorporate a compromise that allows clients to keep open one screen for each injury and solution combination. This would allow multiple injuries and solutions comparisons while minimizing the number of open screens.

The backbone of this application is a Machine Learning method that is human-readable. Part of the GUI development included the desire to give the users as much information as possible to understand how the application provides its decisions. Figure 4.9 is an example of showing the user as much information as possible by showing the user the Decision Tree being used. Dr. Aron's response after the initial proto-type, and the survey confirmed, that showing the Decision Tree is information overload. The students did not understand the information this area of the panel was displaying. They just hit the run button to get a specific score. Future redesign includes reworking the Decision Tree display to enhance it user-friendliness. As well, a default Decision Tree will be loaded automatically rather than forcing a student to load a Decision Tree before running it.

Part of the scenarios asked the students to compare two solutions. In the current configuration the students must open two different solution (Figure 4.12) panels in order to compare the results. The consensus from the comments in the surveys was a desire to automatically run various solution scenarios from the list of injuries given and show the Balance score results in an easy to visualize screen.

Early in discussion with Dr. Aron, there was a desire to implement a module to view medical x-rays. That module was deemed outside the scope of this research. The survey revealed that such a tool for BoneDesktop is highly desired. Many students commented that scoring the attributes would be easier if they could visualize the current injury. Also, recommended was a way to graphically apply fixators to an image of the injury to more accurately assess a proposal.

The scenarios were each designed so that the first scenario went the basic steps from beginning to ending. The other two scenarios used the knowledge garnered from the first scenario to complete the tasks. The users were timed for each scenario they completed. We planned on the first scenario to take 20 minutes to complete while the others were set for 10 minutes each. The first scenario took an average of 27 minutes to complete; more than we planned. The second and third scenario averaged 6 and 9 minutes to completion, respectively. The consensus is that the users did in fact learn how to use the software to complete the assignments; a very positive result.

CHAPTER 6

DECISION TREE AND INSTANCE-BASED LEARNING

The focus of earlier research by Suo [6] revealed that Decision Trees in this application is a very good choice from the various Machine Learning (ML) techniques. Decision tree learning is a method for approximating discrete-value target functions, in which the learned function is represented by the decision tree itself. For this application, most users, including future programmers and the Domain Experts, are not going to have the knowledge nor do they need the knowledge with MLs to further enhance BoneDesktop. The use of Decision Trees facilitates this by representing the Decision Tree knowledge within a set of readable if-then rules as seen in Figure 3.5. The Instance-based learning method we chose to review with this domain data was the K-Nearest Neighbor approach. The K-Nearest Neighbor has the same qualities as the Decision Tree in that it is human readable, robust in its problem solving ability, and is easy to implement.

6.1 Understanding the data

Before reviewing the comparison results we need to understand the data. The current domain is a total of 12 patients. Each patient has one injury where each injury is given 5 different treatments ranging from a poor treatment to an excellent treatment. As described earlier there are 34 parameters for each FAS score and a Balance score defining the value of the proposal. With a large dimensional space (34 attributes) relative to the size of the domain data (60 FAS scores) there is an issue of finding an optimal solution. As the domain size increases we must be concerned with the time taken to classify a new example. Suo's [6] and Aron's [2] research implements a *reduced parameter approach* to help converge on the solution better. The *reduced parameter approach* is a combination of just 4 parameters. Three of the four parameters are the averages of the biological attributes from the FCAS score, the biological attributes from the FTAS score, and the mechanical attributes from the FTAS score. The final attribute is the mean value of the separate averages from the mechanical and clinical attributes.

The Balance score, the resultant value provided by FAS score, is also on a 10 point scale. The values are then externally converted to a 'Poor', 'Good', or 'Excellent' choice where a Balance score of 1 – 4 is poor, 5 – 7 is good, and 8 – 10 is excellent based on Dr. Aron's research [2]. The 'Poor', 'Good', and 'Excellent' choices describe the final ESF proposal based on the given injury. This research presents the idea of converting the 10 point balance score to a ternary decision before developing a Decision Tree rather than after. The goal is to improve the ability to find an acceptable proposal. If a bad proposal is any Balance score from 1 to 4 without concern for how bad the proposal is then a score of 4 is just as bad as a score of 1; a score of 5 has the same value as 7, and a score of 8 is the same as a 10. As with the comparison of a 10 point balance score versus a ternary balance score in the Decision Tree, we want to compare the same results with the K-Nearest Neighbor approach.

6.2 Decision Tree Results

Decision Trees classify instances by sorting them down the tree from some attribute root to a final leaf node attribute, which provides the classification of the instance. Each node in the tree is a simple greater than or less than test of some attribute for the given instance, and each branch descending from that node corresponds to one of the possible values for this attribute. An instance is classified by starting at the root attribute of the tree, then moving down the tree branch corresponding to the found value of the attribute in the given example. This process is then repeated for the subtree rooted at the new node. The original Decision Tree, Iterative Dichotomizer (version) 3 (ID3) [20] is the first algorithm to decide the best attributes currently not in use on which to split the given training data that are classified to the specific branch node. The algorithm flows through the following the steps [22]:

1. If all the instances belong to a single class, there is nothing to do (except create a leaf node labeled with the name of that class).

2. Otherwise, for each attribute that has not already been used, calculate the information gain that would be obtained by using that attribute on the particular set of instances classified to this branch node.

3. Use the attribute with the greatest information gain.

See5 [16] is a software package developed to create a Decision Tree. The See5 Decision Tree family includes the algorithms C4.5, ASSISTANT, and the original ID3 developed by Ross Quinlan [20]. Figure 6.1 represents the configuration possibilities for the See5 application. With a minimal amount of domain data only selective configurations will work. We will review boosting, fuzzy thresholds, and pruning. Cross-validation will help with our comparison by offering realistic random testing for the given data. Boosting generates several classifiers rather than just one. Each new classifier weights the incorrectly defined classes from the previous classifier in determining a Decision Tree. Fuzzy thresholds lessen the drastic change of classification with minor changes in an attribute value. For some domains, a sudden change in classification is necessary – for instance, tax tables have specific cutoff points. It is reasonable to assume that for other domains, a change in classification happens more slowly with attribute value changes. With pruning, a Decision Tree is constructed by See5 in two phases. A large tree is initially grown to closely fit the data. To avoid over fitting the data though, rules predicted to have a high error rate are 'pruned' from the tree.

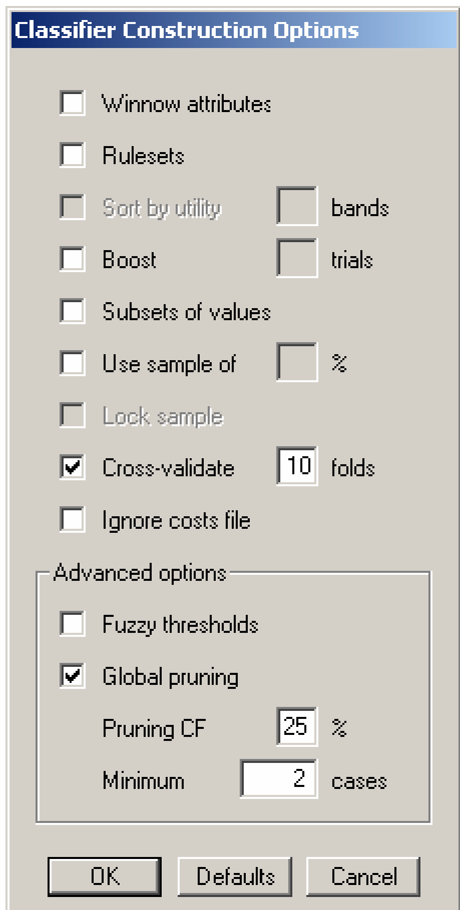

Fig 6.1: See5 Decision Tree Options

Cross-validation is used to help See5 create a situation similar to the real world by randomly pulling data from the data set as target data, while training with the left-over sample data. Due to the random nature of cross-validation the results in Table 6.1 is not representative of an exact figure that will consistently result from test, but a value plus or minus based on the random sampling.

The results from the data demonstrate that the use of pruning, boosting, and using fuzzy thresholds does not significantly improve the predictive model. The current size of the domain data may not allow for the full effect for any one of these configurations. At most the model improves with only a 3.3% increase while the 3 point model actually gets worse with two of the configurations (fuzzy thresholds and pruning). What is important is to notice that the ternary results represent a far superior predictive model than the 10 point scale. The ternary model (without any default configuration options) has a mean error rate of 20% while the 10 point model has a 60% error rate.

Table 6.1: See5 Comparison Between 10 pt. and 3 pt. Balance Score

			Boosting		Fuzzy Thresholds		Pruning	
	10 Point Validation		10 Point Validation		10 Point Validation		10 Point Validation	
Fold	Decision Tree		Decision Tree		Decision Tree		Decision Tree	
----	----------------		----------------		----------------		----------------	
	Size	Errors	Size	Errors	Size	Errors	Size	Errors
0	11	66.7%	*	66.7%	13	33.3%	10	100.0%
1	14	66.7%	*	50.0%	13	100.0%	10	50.0%
2	11	50.0%	*	66.7%	12	66.7%	11	66.7%
3	12	66.7%	*	66.7%	9	66.7%	10	33.3%
4	11	33.3%	*	66.7%	13	50.0%	11	50.0%
5	11	83.3%	*	83.3%	12	66.7%	7	66.7%
6	10	66.7%	*	33.3%	11	66.7%	11	50.0%
7	13	66.7%	*	50.0%	12	50.0%	11	66.7%
8	13	33.3%	*	66.7%	13	33.3%	10	16.7%
9	13	66.7%	*	50.0%	11	50.0%	10	66.7%
----	----------------		----------------		----------------		----------------	
Mean	11.9	60.0%		56.7%	11.9	58.3%	10.1	56.7%
SE	0.4	5.1%		4.4%	0.4	6.2%	0.4	7.1%

			Boosting		Fuzzy Thresholds		Pruning	
	3 Point Validation		3 Point Validation		3 Point Validation		3 Point Validation	
Fold	Decision Tree		Decision Tree		Decision Tree		Decision Tree	
----	----------------		----------------		----------------		----------------	
	Size	Errors	Size	Errors	Size	Errors	Size	Errors
0	9	16.7%	*	16.7%	7	0.0%	7	16.7%
1	9	33.3%	*	16.7%	8	33.3%	8	16.7%
2	7	33.3%	*	0.0%	9	33.3%	9	16.7%
3	5	0.0%	*	50.0%	7	33.3%	7	16.7%
4	7	16.7%	*	16.7%	7	50.0%	7	0.0%
5	8	16.7%	*	16.7%	9	0.0%	9	50.0%
6	9	33.3%	*	16.7%	8	0.0%	8	16.7%
7	9	0.0%	*	16.7%	7	0.0%	7	33.3%
8	7	16.7%	*	33.3%	8	33.3%	8	33.3%
9	9	33.3%	*	16.7%	9	33.3%	9	33.3%
----	----------------		----------------		----------------		----------------	
Mean	7.8	20.0%		20.0%	7.5	21.7%	8.1	23.3%
SE	0.4	4.2%		4.2%	0.4	6.1%	0.3	4.4%

6.3 Instance-based Learning Method

The K-Nearest Neighbor method is the Instance-based Learning technique this study chose to compare with the Decision Tree. The K-Nearest Neighbor predictor uses Euclidean distance with a data set for training to predict the discrete value of each “targeted” attribute. The Euclidean distance between X = (x_1, x_2, x_3,…x_n) and Y = (y_1,y_2, y_3,…y_n) is defined as:

$$D(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Figure 6.2: Euclidean Distance

With the distances defined between a new FAS score and the existing FAS scores the average of the K closest Balance scores is used. The advantages of this ML are its ease of implementation, robustness, and readability. If each new classification is used for future classifications then there are several issues to review. Increasing the data set can increase the program footprint size and will increase the processing time of each a new example. Our research reviewed K values of 1, 2 and 3. Also, with this research, normalizing the data is not needed since each attribute score is within a 10 point range.

The testing of the KNN approach is slightly different than just getting a random sample from the 60 known sets of FAS scores. With only twelve 12 FCAS scores and 5 FTAS proposals for each FCAS score each injury and its 5 proposals must be removed before testing. The KNN approach is based on equal weight among the attributes for each FAS score. If only one FAS score is removed for testing, than there are 4 other FAS score with exactly identical FCAS scores. Allowing these identical scores can grossly affect the performance of the KNN approach by creating bias towards any one of the 4 leftover FAS scores and their respective balance scores. Thus, there are 12 rounds of testing where 5 FAS scores related to each other by identical FCAS scores are tested against the remaining 55 FAS scores.

The results from the comparisons were disappointing, at best. Table 6.2 represents the Euclidean Distances for K = 1. Using the 10 point balance score as the set of target attributes this approach incorrectly finds a target 66.7% of the time. When the ternary balance was introduced it improved to 43.3% error rate. While the ternary scale was vastly improved over the 10 point scale, it is nowhere near currently acceptable. With more domain data the results could very well improve.

For K = 2 and 3 the predictor was worse still. Tables 6.3, 6.4, 6.5, and 6.6 review the results for the K values of 2, 2, 3, and 3, respectively. When the K value is greater than 1, the average of the 2 or 3 closest Balance scores is used to determine the final Balance score. Also the target Balance scores are not discrete values, but continuous where the ceiling of the average Balance scores is used as the comparison. The results did not improve. Using K = 2 and the 3 pt Balance (Table 6.5) score the predictor improved from 43.3% to 38.3% error rate, while the 10 pt Balance score in Table 6.6 shows that the predictor decreases in quality from 66.7.% error rate to a 70% error rate. Using K = 3 did not help the predictor either. Table 6.7 exhibits that the 10 pt Balance score predictor increases horrendously from a 66.7 % error rate to 91.7%. Finally, Table 6.8 does exhibit some slight interest in that the K =3, 3 pt Balance score predictor improves from 43.3% error rate to 41.7% error rate.

Overall, these results demonstrated are inconclusive in determining whether further study with the K-Nearest Neighbor approach is worth continuing. With more domain data, the comparison for the ML techniques can quickly and easily be run again. For this study, though, we chose to proceed strictly with the Decision Tree listed in Appendix C.

Table 6.2: Instance-based Learning Method Comparison Between 10 pt Balance Score and 3pt Balance Score. Highlighted values represent successful search comparison result.

Single Point Values used to find solution.							
FAS Score Name				10 pt. Balance Score		3 pt. Balance Score	
Known	Actual	Target	Distance	Found	66.7%	Distance	43.3%
10A	10	9A	9.798	10	TRUE	3	TRUE
10B	8	9C	9.381	3	FALSE	1	FALSE
10C	5	9C	10.198	3	FALSE	1	FALSE
10D	3	9D	8.185	3	TRUE	1	TRUE
10E	1	9E	7.280	1	TRUE	1	TRUE
11A	10	9B	11.705	6	FALSE	2	FALSE
11B	8	9B	12.610	6	FALSE	2	FALSE
11C	6	15C	11.916	7	FALSE	3	FALSE
11D	5	13D	11.874	3	FALSE	1	FALSE
11E	2	9E	10.536	1	FALSE	1	TRUE
12A	10	7B	11.662	6	FALSE	2	FALSE
12B	6	13B	10.000	6	TRUE	2	TRUE
12C	4	7E	10.630	1	FALSE	1	FALSE
12D	3	7D	11.874	3	TRUE	1	TRUE
12E	1	13E	9.849	1	TRUE	1	TRUE
13A	9	8A	10.247	10	FALSE	3	TRUE
13B	6	14A	9.274	10	FALSE	3	FALSE
13C	3	8D	8.307	3	TRUE	1	TRUE
13D	3	8D	8.944	3	TRUE	1	TRUE
13E	1	14E	8.602	1	TRUE	1	TRUE
14A	10	13B	9.274	6	FALSE	2	FALSE
14B	6	13B	9.747	6	TRUE	2	TRUE
14C	3	13E	9.747	1	FALSE	1	TRUE
14D	2	13C	12.728	3	FALSE	1	TRUE
14E	1	13E	8.602	1	TRUE	1	TRUE
15A	10	16A	10.440	8	FALSE	3	TRUE
15B	8	16B	11.446	6	FALSE	2	FALSE
15C	7	11C	11.916	6	FALSE	2	FALSE
15D	5	10E	10.630	1	FALSE	1	FALSE
15E	2	9E	10.863	1	FALSE	1	TRUE

Single Point Values used to find solution.							
FAS Score Name				10 pt. Balance Score		3 pt. Balance Score	
Known	Actual	Target	Distance	Found	66.7%	Distance	43.3%
16A	8	15A	10.440	10	FALSE	3	TRUE
16B	6	15B	11.446	8	FALSE	3	FALSE
16C	3	18D	11.747	1	FALSE	1	TRUE
16D	2	13D	10.817	3	FALSE	1	TRUE
16E	1	13C	9.381	3	FALSE	1	TRUE
17A	10	9B	11.705	6	FALSE	2	FALSE
17B	7	9B	12.450	6	FALSE	2	FALSE
17C	4	14D	13.038	2	FALSE	1	FALSE
17D	2	14E	12.450	1	FALSE	1	TRUE
17E	1	9E	11.489	1	TRUE	1	TRUE
18A	1	7A	8.832	7	FALSE	3	FALSE
18B	1	7B	9.165	6	FALSE	2	FALSE
18C	1	7C	10.583	5	FALSE	2	FALSE
18D	1	7D	9.165	3	FALSE	1	TRUE
18E	1	7E	10.000	1	TRUE	1	TRUE
7A	7	18A	8.832	1	FALSE	1	FALSE
7B	6	18B	9.165	1	FALSE	1	FALSE
7C	5	18C	10.583	1	FALSE	1	FALSE
7D	3	18D	9.165	1	FALSE	1	TRUE
7E	1	18E	10.000	1	TRUE	1	TRUE
8A	10	13A	10.247	9	FALSE	3	TRUE
8B	6	13B	9.950	6	TRUE	2	TRUE
8C	5	13C	9.644	3	FALSE	1	FALSE
8D	3	13C	8.307	3	TRUE	1	TRUE
8E	1	13E	8.602	1	TRUE	1	TRUE
9A	10	10A	9.798	10	TRUE	3	TRUE
9B	6	10B	10.583	8	FALSE	3	FALSE
9C	3	10B	9.381	8	FALSE	3	FALSE
9D	3	10D	8.185	3	TRUE	1	TRUE
9E	1	10E	7.280	1	TRUE	1	TRUE

Table 6.3: K = 2 For A 3 Pt. Balance Score. The average Balance score helps to slightly increase the predictive quality for this Decision Tree by reducing the error rate from 43.3% to 38.3%.

FAS Score Name		K = 2 Nearest Neighbor, 3 pt. Balance Score									
		First Node				Second Node					
Given	Actual	Target	Distance	Balance	43.3%	Target	Distance	Balance	46.7%	Average	38.3%
10A	3	9A	9.798	3	TRUE	9B	11.747	2	FALSE	2.5	TRUE
10B	3	9C	9.381	1	FALSE	9B	10.583	2	FALSE	1.5	FALSE
10C	2	9C	10.198	1	FALSE	9B	13.153	2	TRUE	1.5	TRUE
10D	1	9D	8.185	1	TRUE	15C	13.000	3	FALSE	2	FALSE
10E	1	9E	7.280	1	TRUE	8E	10.247	1	TRUE	1	TRUE
11A	3	9B	11.705	2	FALSE	9A	11.790	3	TRUE	2.5	TRUE
11B	3	9B	12.610	2	FALSE	16B	13.115	2	FALSE	2	FALSE
11C	2	15C	11.916	3	FALSE	16C	13.191	1	FALSE	2	TRUE
11D	2	13D	11.874	1	FALSE	15D	12.083	2	TRUE	1.5	TRUE
11E	1	9E	10.536	1	TRUE	15E	11.662	1	TRUE	1	TRUE
12A	3	7B	11.662	2	FALSE	16B	12.530	2	FALSE	2	FALSE
12B	2	13B	10.000	2	TRUE	14B	11.533	2	TRUE	2	TRUE
12C	2	7E	10.630	1	FALSE	13C	10.724	1	FALSE	1	FALSE
12D	1	7D	11.874	1	TRUE	7E	12.083	1	TRUE	1	TRUE
12E	1	13E	9.849	1	TRUE	18E	10.050	1	TRUE	1	TRUE
13A	3	8A	10.247	3	TRUE	9B	12.000	2	FALSE	2.5	TRUE
13B	2	14A	9.274	3	FALSE	14B	9.747	2	TRUE	2.5	FALSE
13C	1	8D	8.307	1	TRUE	16E	9.381	1	TRUE	1	TRUE
13D	1	8D	8.944	1	TRUE	16E	10.050	1	TRUE	1	TRUE
13E	1	14E	8.602	1	TRUE	8E	8.602	1	TRUE	1	TRUE
14A	3	13B	9.274	2	FALSE	12B	11.747	2	FALSE	2	FALSE
14B	2	13B	9.747	2	TRUE	12B	11.533	2	TRUE	2	TRUE
14C	1	13E	9.747	1	TRUE	13D	10.770	1	TRUE	1	TRUE
14D	1	13C	12.728	1	TRUE	8C	13.000	2	FALSE	1.5	FALSE
14E	1	13E	8.602	1	TRUE	12E	10.954	1	TRUE	1	TRUE
15A	3	16A	10.440	3	TRUE	9B	11.619	2	FALSE	2.5	TRUE
15B	3	16B	11.446	2	FALSE	16A	12.329	3	TRUE	2.5	TRUE
15C	3	11C	11.916	2	FALSE	16C	11.958	1	FALSE	1.5	FALSE
15D	2	10E	10.630	1	FALSE	13E	10.817	1	FALSE	1	FALSE
15E	1	9E	10.863	1	TRUE	11E	11.662	1	TRUE	1	TRUE
16A	3	15A	10.440	3	TRUE	9B	11.916	2	FALSE	2.5	TRUE
16B	2	15B	11.446	3	FALSE	15A	12.000	3	FALSE	3	FALSE
16C	1	18D	11.747	1	TRUE	15C	11.958	3	FALSE	2	FALSE
16D	1	13D	10.817	1	TRUE	18D	10.909	1	TRUE	1	TRUE
16E	1	13C	9.381	1	TRUE	13D	10.050	1	TRUE	1	TRUE
17A	3	9B	11.705	2	FALSE	9A	11.874	3	TRUE	2.5	TRUE
17B	3	9B	12.450	2	FALSE	14B	13.153	2	FALSE	2	FALSE
17C	2	14D	13.038	1	FALSE	9C	13.528	1	FALSE	1	FALSE
17D	1	14E	12.450	1	TRUE	9E	12.490	1	TRUE	1	TRUE
17E	1	9E	11.489	1	TRUE	8E	11.533	1	TRUE	1	TRUE
18A	1	7A	8.832	3	FALSE	7B	10.488	2	FALSE	2.5	FALSE
18B	1	7B	9.165	2	FALSE	7A	11.136	3	FALSE	2.5	FALSE
18C	1	7C	10.583	2	FALSE	16E	12.806	1	TRUE	1.5	FALSE
18D	1	7D	9.165	1	TRUE	16D	10.909	1	TRUE	1	TRUE
18E	1	7E	10.000	1	TRUE	12E	10.050	1	TRUE	1	TRUE
7A	3	18A	8.832	1	FALSE	18B	11.136	1	FALSE	1	FALSE
7B	2	18B	9.165	1	FALSE	18A	10.488	1	FALSE	1	FALSE
7C	2	18C	10.583	1	FALSE	18D	12.570	1	FALSE	1	FALSE
7D	1	18D	9.165	1	TRUE	12D	11.874	1	TRUE	1	TRUE
7E	1	18E	10.000	1	TRUE	12E	10.247	1	TRUE	1	TRUE
8A	3	13A	10.247	3	TRUE	13B	11.225	2	FALSE	2.5	TRUE
8B	2	13B	9.950	2	TRUE	9C	11.446	1	FALSE	1.5	TRUE
8C	2	13C	9.644	1	FALSE	13D	10.677	1	FALSE	1	FALSE
8D	1	13C	8.307	1	TRUE	13D	8.944	1	TRUE	1	TRUE
8E	1	13E	8.602	1	TRUE	10E	10.247	1	TRUE	1	TRUE
9A	3	10A	9.798	3	TRUE	11A	11.790	3	TRUE	3	TRUE
9B	2	10B	10.583	3	FALSE	15A	11.619	3	FALSE	3	FALSE
9C	1	10B	9.381	3	FALSE	10C	10.198	2	FALSE	2.5	FALSE
9D	1	10D	8.185	1	TRUE	10E	11.533	1	TRUE	1	TRUE
9E	1	10E	7.280	1	TRUE	8E	10.344	1	TRUE	1	TRUE

Table 6.4: K = 2 For A 10 Pt. Balance Score. The average Balance score descreases the predictive quality for this Decision Tree by increasing the error rate from 66.7% from 70.0%.

FAS Score Name		K = 2 Nearest Neighbor, 10 pt. Balance Score									
		First Node				Second Node					
Known	Actual	Target	Distance	Balance	66.7%	Target	Distance	Balance	61.7%	Average	70.0%
10A	10	9A	9.798	10	TRUE	9B	11.747	6	FALSE	8	FALSE
10B	8	9C	9.381	3	FALSE	9B	10.583	6	FALSE	4.5	FALSE
10C	5	9C	10.198	3	FALSE	9B	13.153	6	FALSE	4.5	TRUE
10D	3	9D	8.185	3	TRUE	15C	13.000	7	FALSE	5	FALSE
10E	1	9E	7.280	1	TRUE	8E	10.247	1	TRUE	1	TRUE
11A	10	9B	11.705	6	FALSE	9A	11.790	10	TRUE	8	FALSE
11B	8	9B	12.610	6	FALSE	16B	13.115	6	FALSE	6	FALSE
11C	6	15C	11.916	7	FALSE	16C	13.191	3	FALSE	5	FALSE
11D	5	13D	11.874	3	FALSE	15D	12.083	5	TRUE	4	FALSE
11E	2	9E	10.536	1	FALSE	15E	11.662	2	TRUE	1.5	TRUE
12A	10	7B	11.662	6	FALSE	16B	12.530	6	FALSE	6	FALSE
12B	6	13B	10.000	6	TRUE	14B	11.533	6	TRUE	6	TRUE
12C	4	7E	10.630	1	FALSE	13C	10.724	3	FALSE	2	FALSE
12D	3	7D	11.874	3	TRUE	7E	12.083	1	FALSE	2	FALSE
12E	1	13E	9.849	1	TRUE	18E	10.050	1	TRUE	1	TRUE
13A	9	8A	10.247	10	FALSE	9B	12.000	6	FALSE	8	FALSE
13B	6	14A	9.274	10	FALSE	14B	9.747	6	TRUE	8	FALSE
13C	3	8D	8.307	3	TRUE	16E	9.381	1	FALSE	2	FALSE
13D	3	8D	8.944	3	TRUE	16E	10.050	1	FALSE	2	FALSE
13E	1	14E	8.602	1	TRUE	8E	8.602	1	TRUE	1	TRUE
14A	10	13B	9.274	6	FALSE	12B	11.747	6	FALSE	6	FALSE
14B	6	13B	9.747	6	TRUE	12B	11.533	6	TRUE	6	TRUE
14C	3	13E	9.747	1	FALSE	13D	10.770	3	TRUE	2	FALSE
14D	2	13C	12.728	3	FALSE	8C	13.000	5	FALSE	4	FALSE
14E	1	13E	8.602	1	TRUE	12E	10.954	1	TRUE	1	TRUE
15A	10	16A	10.440	8	FALSE	9B	11.619	6	FALSE	7	FALSE
15B	8	16B	11.446	6	FALSE	16A	12.329	8	TRUE	7	FALSE
15C	7	11C	11.916	6	FALSE	16C	11.958	3	FALSE	4.5	FALSE
15D	5	10E	10.630	1	FALSE	13E	10.817	1	FALSE	1	FALSE
15E	2	9E	10.863	1	FALSE	11E	11.662	2	TRUE	1.5	TRUE
16A	8	15A	10.440	10	FALSE	9B	11.916	6	FALSE	8	TRUE
16B	6	15B	11.446	8	FALSE	15A	12.000	10	FALSE	9	FALSE
16C	3	18D	11.747	1	FALSE	15C	11.958	7	FALSE	4	FALSE
16D	2	13D	10.817	3	FALSE	18D	10.909	1	FALSE	2	TRUE
16E	1	13C	9.381	3	FALSE	13D	10.050	3	FALSE	3	FALSE
17A	10	9B	11.705	6	FALSE	9A	11.874	10	TRUE	8	FALSE
17B	7	9B	12.450	6	FALSE	14B	13.153	6	FALSE	6	FALSE
17C	4	14D	13.038	2	FALSE	9C	13.528	3	FALSE	2.5	FALSE
17D	2	14E	12.450	1	FALSE	9E	12.490	1	FALSE	1	FALSE
17E	1	9E	11.489	1	TRUE	8E	11.533	1	TRUE	1	TRUE
18A	1	7A	8.832	7	FALSE	7B	10.488	6	FALSE	6.5	FALSE
18B	1	7B	9.165	6	FALSE	7A	11.136	7	FALSE	6.5	FALSE
18C	1	7C	10.583	5	FALSE	16E	12.806	1	TRUE	3	FALSE
18D	1	7D	9.165	3	FALSE	16D	10.909	2	FALSE	2.5	FALSE
18E	1	7E	10.000	1	TRUE	12E	10.050	1	TRUE	1	TRUE
7A	7	18A	8.832	1	FALSE	18B	11.136	1	FALSE	1	FALSE
7B	6	18B	9.165	1	FALSE	18A	10.488	1	FALSE	1	FALSE
7C	5	18C	10.583	1	FALSE	18D	12.570	1	FALSE	1	FALSE
7D	3	18D	9.165	1	FALSE	12D	11.874	3	TRUE	2	FALSE
7E	1	18E	10.000	1	TRUE	12E	10.247	1	TRUE	1	TRUE
8A	10	13A	10.247	9	FALSE	13B	11.225	6	FALSE	7.5	FALSE
8B	6	13B	9.950	6	TRUE	9C	11.446	3	FALSE	4.5	FALSE
8C	5	13C	9.644	3	FALSE	13D	10.677	3	FALSE	3	FALSE
8D	3	13C	8.307	3	TRUE	13D	8.944	3	TRUE	3	TRUE
8E	1	13E	8.602	1	TRUE	10E	10.247	1	TRUE	1	TRUE
9A	10	10A	9.798	10	TRUE	11A	11.790	10	TRUE	10	TRUE
9B	6	10B	10.583	8	FALSE	15A	11.619	10	FALSE	9	FALSE
9C	3	10B	9.381	8	FALSE	10C	10.198	5	FALSE	6.5	FALSE
9D	3	10D	8.185	3	TRUE	10E	11.533	1	FALSE	2	FALSE
9E	1	10E	7.280	1	TRUE	8E	10.344	1	TRUE	1	TRUE

Figure 6.5: K = 3 For A 3 Pt. Balance Score. The average Balance score predictor slightly decreases increases in quality as the error rate decreases from 43.3% to 41.7%.

FAS Score Name		K = 3 Nearest Neighbor, 10 pt. Balance Score														
		First Node				Second Node				Third Node						
Known	Actual	Target	Distance	Balance	43.3%	Found	Distance	Balance	46.7%	Target	Distance	Balance	43.3%	Average	41.7%
10A	3	9A	9.798	3	TRUE	9B	11.747	2	FALSE	8A	13.379	3	FALSE	2.667	TRUE
10B	3	9C	9.381	1	FALSE	9B	10.583	2	FALSE	8B	11.790	2	TRUE	1.667	FALSE
10C	2	9C	10.198	1	FALSE	9B	13.153	2	TRUE	13C	14.036	1	FALSE	1.333	TRUE
10D	1	9D	8.185	1	TRUE	15C	13.000	3	FALSE	8D	13.229	1	FALSE	1.667	FALSE
10E	1	9E	7.280	1	TRUE	8E	10.247	1	TRUE	15D	10.630	2	FALSE	1.333	FALSE
11A	3	9B	11.705	2	FALSE	9A	11.790	3	TRUE	15A	12.042	3	TRUE	2.667	TRUE
11B	3	9B	12.610	2	FALSE	16B	13.115	2	FALSE	15B	14.353	3	FALSE	2.333	TRUE
11C	2	15C	11.916	3	FALSE	16C	13.191	1	FALSE	9D	14.142	1	TRUE	1.667	TRUE
11D	2	13D	11.874	1	FALSE	15D	12.083	2	TRUE	13E	12.649	1	FALSE	1.333	TRUE
11E	1	9E	10.536	1	TRUE	15E	11.662	1	TRUE	13D	12.124	1	TRUE	1.000	TRUE
12A	3	7B	11.662	2	FALSE	16B	12.530	2	FALSE	18B	12.610	1	FALSE	1.667	FALSE
12B	2	13B	10.000	2	TRUE	14B	11.533	2	TRUE	14A	11.747	3	FALSE	2.333	FALSE
12C	2	7E	10.630	1	FALSE	13C	10.724	1	FALSE	8C	10.770	2	FALSE	1.333	TRUE
12D	1	7D	11.874	1	TRUE	7E	12.083	1	TRUE	13D	12.490	1	TRUE	1.000	TRUE
12E	1	13E	9.849	1	TRUE	18E	10.050	1	TRUE	7E	10.247	1	TRUE	1.000	TRUE
13A	3	8A	10.247	3	TRUE	9B	12.000	2	FALSE	18A	13.266	1	FALSE	2.000	FALSE
13B	2	14A	9.274	3	FALSE	14B	9.747	2	TRUE	8B	9.950	2	TRUE	2.333	FALSE
13C	1	8D	8.307	1	TRUE	16E	9.381	1	TRUE	8C	9.644	2	FALSE	1.333	FALSE
13D	1	8D	8.944	1	TRUE	16E	10.050	1	TRUE	8C	10.677	2	FALSE	1.333	FALSE
13E	1	14E	8.602	1	TRUE	8E	8.602	1	TRUE	14C	9.747	1	TRUE	1.000	TRUE
14A	3	13B	9.274	2	FALSE	12B	11.747	2	FALSE	8B	12.767	2	TRUE	2.000	FALSE
14B	2	13B	9.747	2	TRUE	12B	11.533	2	TRUE	8B	11.747	2	TRUE	2.000	TRUE
14C	1	13E	9.747	1	TRUE	13D	10.770	1	TRUE	13C	11.269	1	TRUE	1.000	TRUE
14D	1	13C	12.728	1	TRUE	8C	13.000	2	FALSE	17C	13.038	2	TRUE	1.667	FALSE
14E	1	13E	8.602	1	TRUE	12E	10.954	1	TRUE	17E	11.662	1	TRUE	1.000	TRUE
15A	3	16A	10.440	3	TRUE	9B	11.619	2	FALSE	16B	12.000	2	TRUE	2.333	TRUE
15B	3	16B	11.446	2	FALSE	16A	12.329	3	TRUE	9B	12.767	2	FALSE	2.333	TRUE
15C	3	11C	11.916	2	FALSE	16C	11.958	1	FALSE	14C	12.649	1	TRUE	1.333	FALSE
15D	2	10E	10.630	1	FALSE	13E	10.817	1	FALSE	9E	11.045	1	TRUE	1.000	FALSE
15E	1	9E	10.863	1	TRUE	11E	11.662	1	TRUE	16E	11.790	1	TRUE	1.000	TRUE
16A	3	15A	10.440	3	TRUE	9B	11.916	2	FALSE	18A	12.042	1	FALSE	2.000	FALSE
16B	2	15B	11.446	3	FALSE	15A	12.000	3	FALSE	9B	12.369	2	FALSE	2.667	FALSE
16C	1	18D	11.747	1	TRUE	15C	11.958	3	FALSE	13D	12.329	1	FALSE	1.667	FALSE
16D	1	13D	10.817	1	TRUE	18D	10.909	1	TRUE	13C	11.358	1	TRUE	1.000	TRUE
16E	1	13C	9.381	1	TRUE	13D	10.050	1	TRUE	15E	11.790	1	TRUE	1.000	TRUE
17A	3	9B	11.705	2	FALSE	9A	11.874	3	TRUE	14A	13.820	3	TRUE	2.667	TRUE
17B	3	9B	12.450	2	FALSE	14B	13.153	2	FALSE	14D	13.229	1	FALSE	1.667	FALSE
17C	2	14D	13.038	1	FALSE	9C	13.528	1	FALSE	10B	14.036	3	FALSE	1.667	TRUE
17D	1	14E	12.450	1	TRUE	9E	12.490	1	TRUE	14C	12.570	1	TRUE	1.000	TRUE
17E	1	9E	11.489	1	TRUE	8E	11.533	1	TRUE	14E	11.662	1	TRUE	1.000	TRUE
18A	1	7A	8.832	3	FALSE	7B	10.488	2	FALSE	16A	12.042	3	FALSE	2.667	FALSE
18B	1	7B	9.165	2	FALSE	7A	11.136	3	FALSE	16A	12.083	3	TRUE	2.667	FALSE
18C	1	7C	10.583	2	FALSE	16E	12.806	1	TRUE	7D	13.416	1	TRUE	1.333	FALSE
18D	1	7D	9.165	1	TRUE	16D	10.909	1	TRUE	16C	11.747	1	TRUE	1.000	TRUE
18E	1	7E	10.000	1	TRUE	12E	10.050	1	TRUE	13E	10.770	1	TRUE	1.000	TRUE
7A	3	18A	8.832	1	FALSE	18B	11.136	1	FALSE	12A	13.266	3	FALSE	1.667	FALSE
7B	2	18B	9.165	1	FALSE	18A	10.488	1	FALSE	12A	11.662	3	FALSE	1.667	TRUE
7C	2	18C	10.583	1	FALSE	18D	12.570	1	FALSE	12B	13.304	2	FALSE	1.333	TRUE
7D	1	18D	9.165	1	TRUE	12D	11.874	1	TRUE	12C	12.000	2	FALSE	1.333	FALSE
7E	1	18E	10.000	1	TRUE	12E	10.247	1	TRUE	12C	10.630	2	FALSE	1.333	FALSE
8A	3	13A	10.247	3	TRUE	13B	11.225	2	FALSE	12B	12.806	2	TRUE	2.333	TRUE
8B	2	13B	9.950	2	TRUE	9C	11.446	1	FALSE	14B	11.747	2	FALSE	1.667	TRUE
8C	2	13C	9.644	1	FALSE	13D	10.677	1	FALSE	12C	10.770	2	FALSE	1.333	TRUE
8D	1	13C	8.307	1	TRUE	13D	8.944	1	TRUE	13E	10.247	1	TRUE	1.000	TRUE
8E	1	13E	8.602	1	TRUE	10E	10.247	1	TRUE	9E	10.344	1	TRUE	1.000	TRUE
9A	3	10A	9.798	3	TRUE	11A	11.790	3	TRUE	17A	11.874	3	TRUE	3.000	TRUE
9B	2	10B	10.583	3	FALSE	15A	11.619	3	FALSE	17A	11.705	3	TRUE	3.000	FALSE
9C	1	10B	9.381	3	FALSE	10C	10.198	2	FALSE	8B	11.446	2	TRUE	2.333	FALSE
9D	1	10D	8.185	1	TRUE	10E	11.533	1	TRUE	13E	12.247	1	TRUE	1.000	TRUE
9E	1	10E	7.280	1	TRUE	8E	10.344	1	TRUE	11E	10.536	1	TRUE	1.000	TRUE

Figure 6.6: K = 3 For A 10 Pt. Balance Score. The average Balance score predictor decreases horrendously as the error rate jumps from 66.7% to 91.7%.

FAS Score Name		K = 3 Nearest Neighbor, 10 pt. Balance Score													
		First Node				Second Node				Third Node					
Known	Actual	Target	Distance	Balance	66.7%	Actual	Distance	Balance	38.3%	Actual	Distance	Balance	66.7%	Average	91.7%
10A	10	9A	9.798	10	TRUE	9B	11.747	6	FALSE	8A	13.379	10	FALSE	10.582	FALSE
10B	8	9C	9.381	3	FALSE	9B	10.583	6	FALSE	8B	11.790	6	TRUE	6.528	FALSE
10C	5	9C	10.198	3	FALSE	9B	13.153	6	FALSE	13C	14.036	3	FALSE	6.384	FALSE
10D	3	9D	8.185	3	TRUE	15C	13.000	7	FALSE	8D	13.229	3	FALSE	6.333	FALSE
10E	1	9E	7.280	1	TRUE	8E	10.247	1	TRUE	15D	10.630	5	FALSE	5.416	FALSE
11A	10	9B	11.705	6	FALSE	9A	11.790	10	TRUE	15A	12.042	10	TRUE	9.263	TRUE
11B	8	9B	12.610	6	FALSE	16B	13.115	6	FALSE	15B	14.353	8	FALSE	9.038	FALSE
11C	6	15C	11.916	7	FALSE	16C	13.191	3	FALSE	9D	14.142	3	TRUE	7.730	FALSE
11D	5	13D	11.874	3	FALSE	15D	12.083	5	TRUE	13E	12.649	1	FALSE	5.361	FALSE
11E	2	9E	10.536	1	FALSE	15E	11.662	2	TRUE	13D	12.124	3	FALSE	5.221	FALSE
12A	10	7B	11.662	6	FALSE	16B	12.530	6	FALSE	18B	12.610	1	FALSE	6.510	FALSE
12B	6	13B	10.000	6	TRUE	14B	11.533	6	TRUE	14A	11.747	10	FALSE	9.178	FALSE
12C	4	7E	10.630	1	FALSE	13C	10.724	3	FALSE	8C	10.770	5	FALSE	5.575	FALSE
12D	3	7D	11.874	3	TRUE	7E	12.083	1	FALSE	13D	12.490	3	FALSE	6.028	FALSE
12E	1	13E	9.849	1	TRUE	18E	10.050	1	TRUE	7E	10.247	1	TRUE	4.017	FALSE
13A	9	8A	10.247	10	FALSE	9B	12.000	6	FALSE	18A	13.266	1	FALSE	7.667	FALSE
13B	6	14A	9.274	10	FALSE	14B	9.747	6	TRUE	8B	9.950	6	TRUE	8.582	FALSE
13C	3	8D	8.307	3	TRUE	16E	9.381	1	FALSE	8C	9.644	5	FALSE	5.794	FALSE
13D	3	8D	8.944	3	TRUE	16E	10.050	1	FALSE	8C	10.677	5	FALSE	6.017	FALSE
13E	1	14E	8.602	1	TRUE	8E	8.602	1	TRUE	14C	9.747	3	FALSE	4.201	FALSE
14A	10	13B	9.274	6	FALSE	12B	11.747	6	FALSE	8B	12.767	6	TRUE	7.916	FALSE
14B	6	13B	9.747	6	TRUE	12B	11.533	6	TRUE	8B	11.747	6	TRUE	7.844	FALSE
14C	3	13E	9.747	1	FALSE	13D	10.770	3	TRUE	13C	11.269	3	TRUE	4.923	FALSE
14D	2	13C	12.728	3	FALSE	8C	13.000	5	FALSE	17C	13.038	4	FALSE	6.667	FALSE
14E	1	13E	8.602	1	TRUE	12E	10.954	1	TRUE	17E	11.662	1	TRUE	4.318	FALSE
15A	10	16A	10.440	8	FALSE	9B	11.619	6	FALSE	16B	12.000	6	TRUE	8.540	FALSE
15B	8	16B	11.446	6	FALSE	16A	12.329	8	TRUE	9B	12.767	6	FALSE	8.110	FALSE
15C	7	11C	11.916	6	FALSE	16C	11.958	3	FALSE	14C	12.649	3	TRUE	6.986	TRUE
15D	5	10E	10.630	1	FALSE	13E	10.817	1	FALSE	9E	11.045	1	TRUE	4.272	TRUE
15E	2	9E	10.863	1	FALSE	11E	11.662	2	TRUE	16E	11.790	1	FALSE	4.554	FALSE
16A	8	15A	10.440	10	FALSE	9B	11.916	6	FALSE	18A	12.042	1	FALSE	7.639	TRUE
16B	6	15B	11.446	8	FALSE	15A	12.000	10	FALSE	9B	12.369	6	FALSE	8.667	FALSE
16C	3	18D	11.747	1	FALSE	15C	11.958	7	FALSE	13D	12.329	3	FALSE	5.319	FALSE
16D	2	13D	10.817	3	FALSE	18D	10.909	1	FALSE	13C	11.358	3	FALSE	5.636	FALSE
16E	1	13C	9.381	3	FALSE	13D	10.050	3	FALSE	15E	11.790	2	FALSE	5.017	FALSE
17A	10	9B	11.705	6	FALSE	9A	11.874	10	TRUE	14A	13.820	10	TRUE	9.291	TRUE
17B	7	9B	12.450	6	FALSE	14B	13.153	6	FALSE	14D	13.229	2	FALSE	7.051	FALSE
17C	4	14D	13.038	2	FALSE	9C	13.528	3	FALSE	10B	14.036	8	FALSE	7.843	FALSE
17D	2	14E	12.450	1	FALSE	9E	12.490	1	FALSE	14C	12.570	3	FALSE	5.497	FALSE
17E	1	9E	11.489	1	TRUE	8E	11.533	1	TRUE	14E	11.662	1	TRUE	4.511	FALSE
18A	1	7A	8.832	7	FALSE	7B	10.488	6	FALSE	16A	12.042	8	FALSE	8.496	FALSE
18B	1	7B	9.165	6	FALSE	7A	11.136	7	FALSE	16A	12.083	8	FALSE	8.379	FALSE
18C	1	7C	10.583	5	FALSE	16E	12.806	1	TRUE	7D	13.416	3	FALSE	6.935	FALSE
18D	1	7D	9.165	3	FALSE	16D	10.909	2	FALSE	16C	11.747	3	FALSE	5.636	FALSE
18E	1	7E	10.000	1	TRUE	12E	10.050	1	TRUE	13E	10.770	1	TRUE	4.017	FALSE
7A	7	18A	8.832	1	FALSE	18B	11.136	1	FALSE	12A	13.266	10	FALSE	7.379	FALSE
7B	6	18B	9.165	1	FALSE	18A	10.488	1	FALSE	12A	11.662	10	FALSE	7.163	FALSE
7C	5	18C	10.583	1	FALSE	18D	12.570	1	FALSE	12B	13.304	6	FALSE	6.523	FALSE
7D	3	18D	9.165	1	FALSE	12D	11.874	3	TRUE	12C	12.000	4	FALSE	5.625	FALSE
7E	1	18E	10.000	1	TRUE	12E	10.247	1	TRUE	12C	10.630	4	FALSE	5.082	FALSE
8A	10	13A	10.247	9	FALSE	13B	11.225	6	FALSE	12B	12.806	6	TRUE	8.742	FALSE
8B	6	13B	9.950	6	TRUE	9C	11.446	3	FALSE	14B	11.747	6	FALSE	7.815	FALSE
8C	5	13C	9.644	3	FALSE	13D	10.677	3	FALSE	12C	10.770	4	FALSE	5.892	FALSE
8D	3	13C	8.307	3	TRUE	13D	8.944	3	TRUE	13E	10.247	1	FALSE	4.315	FALSE
8E	1	13E	8.602	1	TRUE	10E	10.247	1	TRUE	9E	10.344	1	TRUE	4.082	FALSE
9A	10	10A	9.798	10	TRUE	11A	11.790	10	TRUE	17A	11.874	10	TRUE	10.597	FALSE
9B	6	10B	10.583	8	FALSE	15A	11.619	10	FALSE	17A	11.705	10	TRUE	9.873	FALSE
9C	3	10B	9.381	8	FALSE	10C	10.198	5	FALSE	8B	11.446	6	FALSE	8.066	FALSE
9D	3	10D	8.185	3	TRUE	10E	11.533	1	FALSE	13E	12.247	1	TRUE	5.178	FALSE
9E	1	10E	7.280	1	TRUE	8E	10.344	1	TRUE	11E	10.536	2	FALSE	4.448	FALSE

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

BoneDesktop focused on the development of a visualization tool, but involved a broad array of processes, review, and knowledge necessary to properly develop a beta release as the beginning of the software life-cycle. The early work with Dale [4] and Suo [6] helped to lay a foundation with a direction towards the best Machine Learning techniques to use with the domain data. Their work did not consider the true users of the application. The development of BoneDesktop is based on a user-centered approach. By applying an early focus on the users and what they were trying to complete, reviewing early proto-types, and iterating the design BoneDesktop has strived to meet the goals of Dr. Aron.

The long-term success of BoneDesktop is grounded in a framework design that helps other programmers quickly learn the code. Utilizing proven design patterns and existing open source projects helps to ensure that over time the peripheral technologies implemented will remain stable.

User feed-back for the early beta version was very positive, but it revealed key areas that needed more work. The MDI interface was implemented to better separate content into common groupings. Unfortunately the BoneDesktop screen can get cluttered if not controlled. A current redesign to focus an injury and its solutions into one internal frame is being tested. The redesign still separates the content, but minimizes the number of open panels.

Building a tool focused on long-term development and use was a major goal for this study. BoneDesktop is not a throw away prototype, but a foundation for future work. In order to continue with the development of BoneDesktop the following points will need to be addressed:

- Unit testing is required to remove any subtle bugs that are in the application.
- With the amount of domain data available the current work with the Decision Tree is focused on the qualities of this type of data. More Domain Data needs to be recorded allowing the Decision Tree to improve.
- In some situations not all of the attributes are required in the scoring of an injury. A set of rules for controlling the use of attributes is required.
- While BoneDesktop can be run in a networked environment, it is not synchronized to do so. Re-factoring the code for safe use on a network is a big step towards scaling up the amount of domain data available.

REFERENCES

[1] College of Veterinary Medicine, University of Florida [Online]. [Accessed 1st April 2005]. Available from World Wide Web: <http://www.vetmed.ufl.edu/sacs/Lewis/External_Fixation/page2.html>

[2] The Fracture-Patient Assessment Score (FPAS), Dennis N. Aron. Proceedings of the 12th Annual Complete Course in External Skeletal Fixation, Ames Iowa, September 10-12, 2004.

[3] Potter W. D. Bone-Fixes: An external skeletal fixation expert system. In The 2000 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences, Las Vegas, 2000.

[4] Duda, Richard and Shortliffe Edward (1983) Expert systems research. Science. 200(4594): 261-268.

[5] Dale Jr., Harold Edward. Bone-Fixes: An expert system for external skeletal fixation. Master's thesis, The University of Georgia, 2000.

[6] Sing, Nuo. Machine Learning Techniques for the Evaluation of External Skeletal Fixation Structures. Master's thesis, The University of Georgia, 2003.

[7] Reed, Paul R. Developing UML Application with Java. New York: Addison-Wesley, 2002.

[8] Open Source Initiative [Online]. [Accessed 15th March 2005]. Available from World Wide Web: <http://www.opensource.org/>

[9] DuBois, Paul. MySQL Cookbook. Cambridge: O’reilly and Associates, 2002.

[10] World Wide Web Consortium [Online]. [Accessed 4th December 2004]. Available from World Wide Web: <http://www.w3.org/XML/>

[11] Kuchana, Partha. Software Architecture Design Patterns in Java. New York: Auerbaach Publications, 2004.

[12] JScroll – The Scrollable Desktop Pane for Java [Online]. [Accessed May 10th April 2004]. Available from World Wide Web: < http://jscroll.sourceforge.net/>

[13] Preece, Jenny, and Rogers, Yvonne, and Sharp, Helen. Interaction Design. New York City: John Wiley & Sons, Inc, 2002.

[14] Design Guidelines: Behavior [Online] [Accessed 6th April. 2005] Available from World Wide Web: <http://java.sun.com/products/jlf/ed1/dg/higi.htm#35796>

[15] Internationalization [Online] [Accessed 10th May 2004] Available from the World Wide Web <http://java.sun.com/docs/books/tutorial/i18n/>

[16] Information On See5 / C5.0 [Online] [Accessed 20th November 2004] Available from the World Wide Web <http://www.rulequest.com/see5-info.html>

[17] Apache Ant [Online] [Accessed 8th May 2003] Available from the World Wide Web <http://ant.apache.org/>

[18] xml.apache.org [Online] [Accessed 28th May 2004] Available from the World Wide Web <http://xml.apache.org/>

[19] Duda, R.O. and E.H. Shortliffe (1983) Expert systems research. Science. 220(4594): 261-268.

[20] Quinlan, Ross. Discovering Rules by Induction From Large Collections of Examples. In D. Michie (Ed.), Expert Systems in the micro electronic age. Edinburgh University Press, 1979.

[21] Mitchell, Tom. Machine Learning, New York City: WCB/McGraw Hill, 1997.

[22] The Machine Learning Dictionary [Online] [Accessed 15th April 2005] Available from the World Wide Web http://www.cse.unsw.edu.au/~billw/mldict.html>

APPENDIX A

BONEDESKTOP DATABASE SQL STATEMENTS

Appendix A is the current set of SQL statements needed to run the BoneDesktop application.

```
drop database doggydb;

create database doggydb;

use doggydb;

create table people(
    ownerId int not null auto_increment,
    firstname varchar(40),
    lastname varchar(40),
    address1 varchar(100),
    address2 varchar(100),
    city varchar(40),
    state varchar(40),
    zip varchar(10),
    phone varchar(10),
    datelastmodified date,
    primary key(ownerId));

create table animals(
    petId int not null auto_increment,
    name varchar(40),
    birthdate date,
    deathdate date,
    breed varchar(20),
    fixed int,
    gender varchar(20),
    weight varchar(3),
    weightLabel varchar(10),
    dateentered date,
    datelastmodified date,
    primary key(petId));

create table ownedanimals(
    petId int not null,
    ownerId int not null,
    primary key(petId,ownerId));

create table injuries(
    injuryId int not null auto_increment,
    shortdescription varchar(255),
    history MEDIUMTEXT,
    character_ MEDIUMTEXT,
    generalcondition MEDIUMTEXT,
    lameness MEDIUMTEXT,
    softtissue MEDIUMTEXT,
    fracture MEDIUMTEXT,
    neurological MEDIUMTEXT,
    dateentered date,
    datelastmodified date,
    MAC1 int DEFAULT '0',
    MAC2 int DEFAULT '0',
    MAC3 int DEFAULT '0',
    MAC4 int DEFAULT '0',
    MAC5 int DEFAULT '0',
    MAC6 int DEFAULT '0',
    BAC1 int DEFAULT '0',
    BAC2 int DEFAULT '0',
    BAC3 int DEFAULT '0',
    BAC4 int DEFAULT '0',
    BAC5 int DEFAULT '0',
    BAC6 int DEFAULT '0',
    BAC7 int DEFAULT '0',
    BAC8 int DEFAULT '0',
    BAC9 int DEFAULT '0',
    BAC10 int DEFAULT '0',
    CAC1 int DEFAULT '0',
    CAC2 int DEFAULT '0',
```

```
    CAC3 int DEFAULT '0',
    primary key(injuryId));

create table solutions(
    solutionId int not null
auto_increment,
    injuryId int,
    dateentered date,
    datelastmodified date,
    name varchar(50),
    MAC1 int DEFAULT '0',
    MAC2 int DEFAULT '0',
    MAC3 int DEFAULT '0',
    MAC4 int DEFAULT '0',
    MAC5 int DEFAULT '0',
    MAC6 int DEFAULT '0',
    MAC7 int DEFAULT '0',
    MAC8 int DEFAULT '0',
    MAC9 int DEFAULT '0',
    MAC10 int DEFAULT '0',
    MAC11 int DEFAULT '0',
    MAC12 int DEFAULT '0',
    BAC1 int DEFAULT '0',
    BAC2 int DEFAULT '0',
    BAC3 int DEFAULT '0',
    BAC4 int DEFAULT '0',
    BAC5 int DEFAULT '0',
    BAC6 int DEFAULT '0',
    primary key(solutionId));

create table animalinjuries(
    petId int not null,
    injuryId int not null,
    primary key(petId,injuryId));
```

APPENDIX B

USER SURVEY

Survey questionnaires for BoneDesktop Application:

1. What is your current level in the veterinarian program?

_____Student _____Intern _____ Resident

2. How comfortable are you with using computer software?

_____Very comfortable _____Comfortable _____Neutral _____Uncomfortable _____Very Uncomfortable

3. Have you ever used or reviewed a program with a similar application of Artificial Intelligence:

_____Yes _____No

4. How did you find the User Interface?

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

Is there any specific item about the software that stands out either positively or negatively.

5. Did you find any of the Label names or Screens confusing? _____Yes _____No.

If yes, please mention the labels or screens that were confusing.

6. Did the overall design have the same feel has other windows based programs.

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

If poor, please mention what didn't 'feel' right.

7. Was the overall design easy to use? _____Yes _____No.

If no, please suggest areas that caused problems or confusion.

8. Please rate the flow of the program on ease of navigation:

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

9. Please grade the program on overall look:

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

10. The vocabulary on the program is appropriate for the intended audience:

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

11. The text is clearly written and easy to understand:

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

12. The information needed for the intended audience is sufficient.

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

If this program is missing some specific piece of information please note it here:

13. The organization of the program is logical and clear.

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

14. The size of the text is easy to read.

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

15. How easy was it to maneuver the injury or search results panel to review the injuries or solutions.

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

16. Were the slider features useful in quantifying a specific value for an injury or solution:

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

If poor, please offer a suggestion for improving how to quantify specific element of an injury or a solution:

17. Was the summary tab on the solutions frame easy to understand?

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

If Poor, do you have any suggestions for improving the understanding of it:

18. Rate how usable you feel the software, in its current state, would aid in helping learning best scenarios for applying a fixator:

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

What would you consider is needed in order to improve its ability at helping to learn best scenarios for applying a fixator:

19. Rate how usable you feel the software, it its current state, would aid in helping to confirm a doctor's choice in applying a fixator:

_____Excellent _____Good _____Neutral _____Satisfactory _____Poor

What would you consider is needed in order to improve its ability at helping to learn best scenarios for applying a fixator:

20. Please offer any other ideas for the program:

APPENDIX C

SEE5 OUTPUT DECISION TREE USED FOR TESTING

```
See5 [Release 1.20a]   Tue Feb 22 23:54:55 2005
--------------------

Read 60 cases (4 attributes) from dog_orig_ave.data

Decision tree:

MechanicalTreatmentScoringAverage > 5.91:
:...BiologicalTreatmentScoringAverage > 5.17: 10 (14/5)
:   BiologicalTreatmentScoringAverage <= 5.17:
:   :...BiologicalInjuryScoringAverage <= 3.4: 8 (2)
:       BiologicalInjuryScoringAverage > 3.4: 6 (8/3)
MechanicalTreatmentScoringAverage <= 5.91:
:...MechanicalTreatmentScoringAverage > 4.3:
    :...MechanicalClinicalInjuryScoringAverage <= 5.1: 5 (7/3)
    :   MechanicalClinicalInjuryScoringAverage > 5.1:
    :   :...BiologicalInjuryScoringAverage <= 4.25: 4 (2)
    :       BiologicalInjuryScoringAverage > 4.25: 3 (5)
    MechanicalTreatmentScoringAverage <= 4.3:
    :...MechanicalClinicalInjuryScoringAverage <= 3.9: 2 (3/2)
        MechanicalClinicalInjuryScoringAverage > 3.9:
        :...MechanicalTreatmentScoringAverage > 3.82: 2 (3)
            MechanicalTreatmentScoringAverage <= 3.82:
            :...MechanicalTreatmentScoringAverage <= 3: 1 (8/1)
                MechanicalTreatmentScoringAverage > 3:
                :...BiologicalTreatmentScoringAverage <= 3: 1 (4/1)
                    BiologicalTreatmentScoringAverage > 3: 3 (4)


Evaluation on training data (60 cases):

            Decision Tree
            ----------------
            Size      Errors

            11   15(25.0%)   <<
```

(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)	(j)	<-classified as
----	----	----	----	----	----	----	----	----	----	
10										(a): class 1
1	4									(b): class 2
1		9		1						(c): class 3
			2	1						(d): class 4
	1			4						(e): class 5
	1				5				2	(f): class 6
				1	2					(g): class 7
							2		3	(h): class 8
					1					(i): class 9
									9	(j): class 10

Time: 0.0 secs

APPENDIX D

UPDATING DECISION TREE, FAS ATTRIBUTES IN APPLICATION, AND DATABASE

One of the ongoing goals of this project is to collect more domain data. With new data a new learning tree needs development. This appendix demonstrates the following two possibilities:

1. New data has been collected and a new decision tree needs development.
2. A change in the FAS attributes has occurred. The application needs updating.

The following assumptions are made: the See5 program is used to develop the decision tree and the database in use is MySQL.

When developing the new decision tree with See5 two files are, at a minimum, required. The .names file describes the classification values as well as the attributes used for classifying. Figure A.1 is an example of the text file.

```
                                                 | Names file for averages
1,2,3,4,5,6,7,8,9,10                             | classes to be predicted
                                                 |
BiologicalInjuryScoringAverage:        continuous.| Averages for Biological Injury Attributes
BiologicalTreatmentScoringAverage:     continuous.| Averages for Biological Treatment Attributes
MechanicalClinicalInjuryScoringAverage:continuous.| Averages for Mechanical,Clinical Injury Attributes
MechanicalTreatmentScoringAverage:     continuous.| Averages for Biological Treatment Attributes
```

Figure A.1: A *.names File For See5

The second file containing the new injury and proposal attributes is the .data file. Each line in the file is a complete injury and proposal data combination. The values are comma de-limited

```
2.85, 5.25, 5.65, 8.18, 10
2.85, 4.75, 5.65, 6.91, 8
5.2, 3.67, 4, 4.64, 5
5.2, 3.17, 4, 3.45, 3
5.2, 2.83, 4, 2.09, 1
7.14, 8, 4.83, 8.45, 10
7.14, 6.75, 4.83, 7.09, 6
```

Figure A.2: A *.data File For See5

and the last value in the set must be the class prediction value. Figure A.2 is an example of what the file might look like. Notice the first four values from Figure A.2 correspond to the 'Averages for the Biological Injury Attributes', 'Averages for the Biological Treatment Attributes', 'Averages for the Mechanical, Clinical Injury Attributes', and 'Averages for the Mechanical Treatment Attributes.' The fifth value is the Balance score prediction for those attributes.

The two files, .data and .names files, must be in the same directory in order for See5 to properly work. Once the files are loaded and See5 is running then locate the data file. Users knowledgeable with the configuration options with See5 can adjust them as necessary to look for improvements with the Decision Tree. Using just the default configurations, click the 'Construct Classifier' button, then click 'Ok'. The classifier is now constructed in the folder where the .data file and .names file are located. Users will choose this new rule set on the 'Summary' tab of the 'Enter Solutions Panel'. The rule set will load and users will click the 'Run' button.

If attributes need adjustment such as adding, rearranging, or deleting them, then this is a simple procedure. Locate the resources folder within the BoneDesktop folder. In the resources folder there is a file titled ScoringProperties.xml. The basic structure of an XML file is a tree structure where each node is inside another node except for the root node. Each of the five categories is listed with the attributes that describe them. Notice that the first attribute under the clinical node is 'Patient' with an order of '0' where 0 is the first node in order, '1' is the second node, and so on. The second node is 'Client' with an order of '1'. By switching the '0' and '1' values the order in which ‘Client” and Patient’ are shown in the application. By creating a new attribute value using the ‘Patient’ node as an example and setting the order to ‘2’ another node is added.

There are some specific rules that must be followed. It is important to start the order for each category with’0’ and sequentially continue for the rest of the nodes. 10 value nodes are required for each attribute. When these changes are completed, the database will need updating to reflect the updated xml file.

When attributes are changed in the ScoringProperties.xml file, the database portion also needs updating. The attributes in the database need to be ordered in the same fashion as the ScoringProperties.xml file. The three types of attributes in the two tables are BAC#, CAC#, and MAC# where each value is biological, clinical, or mechanical, respectively, and the # represents the order number in the ScoringProperties.xml file plus 1 (i.e., if the order number is zero then the # is 1).

Assuming the use of MySQL the ALTER TABLE command is the proper syntax. When adding a new attribute to the end of the table, use the following syntax:

ALTER TABLE <table name> ADD <BAC#/CAC#/MAC#> int

To add an attribute after an existing column use the following syntax:

ALTER TABLE <table name> ADD <BAC#/CAC#/MAC#> int AFTER <column name>

When removing an attribute use the following syntax:

ALTER TABLE <table name> DROP <column name>

It is important to note that when a column is dropped all data in that column is lost.

If a column is reordered then use:

ALTER TABLE modify <column name> int AFTER <column name>

Inserting the new data is best done through the interface so that the safety checks are used to make sure the data is not corrupt when inserted. Experienced users can use the INSERT VALUE syntax to quickly add the new information, although, in order to discourage that, this thesis will not offer an example on that procedure.