Project Number: 51-PWD-6022

Educational Software Help System

DElab Help System

An Interactive Qualifying Project Report
submitted to the Faculty of
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
Degree of Bachelor of Science
by

Alex Sanville
(sanville@wpi.edu) _____

Cyril Shilnikov
(shilkand@wpi.edu) _____

Stefan Slonevskiy
(slonsl@wpi.edu) _____

WPI Advisor:
Professor Paul W. Davis
(pwdavis@wpi.edu) _____

Worcester Polytechnic Institute
100 Institute Road Worcester, MA 01609

Abstract

This project developed an improved help system for DELab, a MATLAB-based educational software package for introductory differential equations. The project team designed a framework for future implementation that integrated concepts from the education literature, experiences of expert teachers, the latest principles of human-computer interaction , and today's best help system practices as exhibited in comparable software. The team provided a sample implementation and guidelines for a future complete implementation.

Executive Summary

Many of the math classes at WPI rely on some form of mathematical software to supplement the course. Mathematical software allows students to perform complex operations that would be difficult to do by hand and has become an indispensable part of the WPI curriculum. One such software is our particular area of interest. DElab is a mathematical software used in some of the differential equations courses taught at WPI.

The main issue we addressed with this project was to compare the DElab help system to today's standards for help systems and then design a basis for a new help system that would meet or exceed these standards. Updating the help system would have many benefits. It would allow students to use DElab more efficiently, aid in the understanding of differential equations, and provide professors with a tool that would need minimal explanation.

Through reviewing the literature on the subject and consulting experts we have found that, as initially expected, the old help system is inadequate by today's standards. The old help system consists mostly of text files. These text files contain valuable information but lack key components that make up a good help system. The current help system lacks search capability, index, graphics of any kind, organization, consistency, conceptual information, and many other components. Research has shown us that all these components play an important role in the effectiveness of a help system. The new design takes each of these components into account and the end result is a help system that rivals the best help systems on the market.

In our research on learning and help systems there were some questions that arose that we needed to answer. Should we include conceptual information on the particular

function? Is it necessary to inform the user of what the program is doing? Does the way the help system presents information really matter? These types of high level questions need to be answered so that all the details of the help system could be filled in. Understanding what a good help system contains provides a framework that can be applied to any help system.

Once this framework is created it then becomes possible to fill it out with information relevant to the area of interest. When filling out the framework of the help system everything from font size to information detail needs to be considered. Even though there are not set standards for doing this, there are some common practices that should be used. These common practices may vary greatly depending on the type of help system being designed. When the framework is completely filled out it becomes a design for a new help system that is appropriate and unique to its function.

This project built such a framework beginning with a literature review, then filling out this framework with information obtained from interviews of appropriate experts and evaluations of help systems in comparable software. This project also details the implementation of the design. Time restraints did not permit us to fully implement the help system. Only the main structure has been implemented. Our team also created help pages on a single software function of DElab as an example for those who fully implement the help system. As an end result this project has discovered what the major building blocks of a good help system are and how to use them in creating the best possible help system for the particular package DElab. This new help system, when fully implemented, should make learning differential equations easier for the students at WPI.

The previous paragraph briefly mentioned the current state of the help system. The implementation stage was done on a smaller scale with only one software function explained. There are a number of items that must be dealt with before the system can be released to the public. These items vary from very minor changes of the interface text to major changes such as new text for the main sections of the help system. Most notably the unwritten text for the rest of the software functions must be written. Bugs in the DElab software that were detected during implementation stage must be fixed. A list of common mistakes for use in a Common Errors section must be compiled. And detailed installation instructions for non-Windows operating systems must be written.

The list of items that still need to be completed is relatively short in comparison to the list of concrete accomplishments. Our team was able to integrate the new help system into the MATLAB environment in a seamless fashion. The DElab help section now appears in MATLAB's table of contents along with other MATLAB help topics. DElab can also be searched through the index function provided by MATLAB's help system. Our team also gathered enough information to implement a search function. One section of the new help system has been implemented as an example of the formatting used. Based on this partial implementation the rest of the help system design is relatively painless.

Also, to address some of the unexpected issues such as the difficulty of the installation process, our team designed an automatic installer for DElab core files and help files. The installer includes an uninstall function to ensure the integrity of the system after the DElab has been removed. The installer can be used as is, without any modifications.

The research based on the examples of the most successful mathematical packages allowed the team to add information to many areas which were under-researched in the original DElab help system. This ensures that the work that was done corresponds to the industry standards that are in effect now. Most users are familiar with some of the software on the market today. Having a system that follows the same standards provides the user with a certain comfort level and allows them to easily adapt to the DElab software.

The purpose of the project was to create a new help system design with a user in mind was successfully accomplished. The final product that the team produced is a detailed description of a new help system with examples that demonstrate the ideas taken from the research and interviews. With this framework established, it is only a matter of time before the entire help system can be implemented.

Table of Contents

Table of Contents i
Table of Figures iii
1. Introduction 1
2. Literature Review 3
2.1 Learning Differential Equations 3
2.2 Teaching Mathematics with Aid of Computer Software 6
2.3 Help File Components 9
2.4 Other Considerations 13
2.5 Learning Mathematics and the Help System 14
3. Methodology 18
3.1 What Content Should the Help System Provide? 18
3.2 Conducting Interviews 20
3.2.1 How Should the Help System be Organized? 22
3.2.2 What Should the Help System Have for a User Interface? 24
4. Data and Analysis 27
4.1 Interviews 27
4.1.1 Differential Equations Professor Interviews 27
4.1.2 Professor Heinricher 27
4.1.3 Professor Fehribach 29
4.1.4 Analysis of Differential Equations Professor Interviews 31
4.1.5 Professor Brown 33
4.1.6 Analysis of HCI Principles 35
4.2 Software Evaluation 36
4.2.2 Analysis of the Value Matrix. 38
4.3 DElab Evaluation 42
4.4 Overall Analysis 44
4.4.1 Items to be Added 44
4.4.2 Items to be Changed 48
4.4.3 Items to be Removed 49
5. Implementation 51
5.1 Implementation Plan 51
5.2 Obstacles Encountered 54
5.3 Accomplishments 55
5.3.1 Direction Fields 55
5.3.2 HCI Principles 59
5.4 Technical Specifications 62
5.4.1 Help Structure 62
5.4.2 Overall Layout 63
5.4.3 Table of Contents 64
5.4.4 Index 64
5.4.5 Search 65
6. Conclusions and Recommendations 66
6.1 Further Steps 66

6.2 Final Thoughts 69
Bibliography 71
Appendix A: Interview Questions 73
Appendix B: Professor Heinricher 74
Appendix C: Professor Fehribach 76
Appendix D: Professor Brown 77
Appendix E: Value Matrix 79

Table of Figures

Figure 5.1: “Background Information” Window 57
Figure 5.2: Using the “Direction Fields Window” Help Screen 57
Figure 5.3: “Interpreting Results” Window 58
Figure 5.4: “Common Errors” Screen 58
Figure 5.5: Table of Contents 60
Figure 5.6: Chapter 1 Examples 61
Figure 5.7: Index 63

1. Introduction

Many of the math classes at WPI rely on some form of mathematical software to supplement the course. Mathematical software allows students to perform complex operations that would be difficult to do by hand and has become an indispensable part of the WPI curriculum. One such software is our particular area of interest. DElab is a mathematical software used in some of the differential equations courses taught at WPI.

The main issue we addressed with this project was to compare the DElab help system to today's standards for help systems and then design a basis for a new help system that would meet or exceed these standards. Updating the help system would have many benefits. It would allow students to use DElab more efficiently, aid in the understanding of differential equations, and provide professors with a tool that would need minimal explanation.

Through reviewing the literature on the subject and consulting experts we have found that, as initially expected, the old help system is inadequate by today's standards. The old help system consists mostly of text files. These text files contain valuable information but lack key components that make up a good help system. The current help system lacks search capability, index, graphics of any kind, organization, consistency, conceptual information, and many other components. Research has shown us that all these components play an important role in the effectiveness of a help system. The new design takes each of these components into account and the end result is a help system that rivals the best help systems on the market.

In our research on learning and help systems there were some questions that arose that we needed to answer. Should we include conceptual information on the particular

function? Is it necessary to inform the user of what the program is doing? Does the way the help system presents information really matter? These types of high level questions need to be answered so that all the details of the help system could be filled in. Understanding what a good help system contains provides a framework that can be applied to any help system.

Once the framework is created it then becomes possible to fill it out with information relevant to the area of interest. When filling out the framework of the help system everything from font size to information detail needs to be considered. Even though there are not set standards for doing this, there are some common practices that should be used. These common practices may vary greatly depending on the type of help system being designed. When the framework is completely filled out it becomes a design for a new help system that is appropriate and unique to its function.

This project goes through this process of building a framework through literature review and then fills this framework out with information obtained from methodology. This project also details the implementation of the design. Time restraints did not permit us to fully implement the help system. Only the main structure has been implemented. Our team also created help pages on a single software function of DElab as an example for those who fully implement the help system. As an end result this project has discovered what the major building blocks of a good help system are and how to use them in creating the best possible help system for the particular package DElab. This new help system, when fully implemented, should make learning differential equations easier for the students at WPI.

2. Literature Review

The information presented in this chapter provides a rough framework of the ways in which help files for DElab can be improved. The three major aspects that determine the usefulness of these help files are the relevance to the differential equations curriculum, the best practices of using computer software in teaching class material, and the best ways of organizing and structuring help files themselves. Chapter 4 reviews and explores the aforementioned topics in depth and fills out the help system framework provided by this chapter. Chapter 4 also connects these topics to the DElab help system.

2.1 Learning Differential Equations

People learn mathematics and in particular differential equations in a number of ways. Some of the theories connected with learning examine the student's role in learning and others examine learning as it pertains to the teacher. Understanding the teacher-student relationship is important when dealing with any subject in education. There have been many attempts to model the student teacher relationship. With time these theories about the student-teacher relationship have evolved and changed greatly. Despite the change, each theory has certain concepts that still hold true today.

Connectionism was an early theory of about how students learn mathematics. Simply stated Connectionism states that students learn through a system of repetition and reward. (Selden & Selden 1993) This is still true today; much of learning is based on repetition and reward. School work is often repetitive in nature and this allows the students to practice and master a certain concept. The reward is the grade that the

students receive from the work they have done. This theory is effective for teaching complex operations that can only be mastered through repetition. However this theory of teaching often neglects the conceptual side of mathematics. That is that Connectionism style learning does not always look at the meaning behind a mathematical process. The meaning or concept behind each mathematical process is just as important as the process itself. Concepts are the tools that help math professors remember all of the mathematical processes. It is said that mathematical professors do not remember all of the complex processes. Instead they understand the concepts which drive these processes and this allows them to recreate the processes without memorization. (Selden & Selden 1993)

A more relevant theory on learning mathematics is Constructivism. Constructivism can be defined as a

> "…a philosophy of learning founded on the premise that, by reflecting on our experiences, we construct our own understanding of the world we live in. Each of us generates our own "rules" and "mental models," which we use to make sense of our experiences. Learning, therefore, is simply the process of adjusting our mental models to accommodate new experiences." (Funderstanding.com 1998)

Unlike learning theories before it Constructivism portrays the student as an active participant in the learning process. Other theories related the student to a blank slate or an empty vessel. This is not an accurate model of a student. In many cases students may have prejudices or pre-conceived ideas about the subject matter. Constructivism remains as one of the most "influential and widely accepted philosophical perspective in mathematics." (Selden & Selden 1993)

The principles of Constructivism can be seen throughout almost any curriculum in the country. One of these principles deals specifically with how material should be organized so that it can be more easily learned by the student. This principle states that it

is best to teach using building blocks, so that each new concept builds upon the last. (Funderstading.com 1998) Another point that Constructivism makes is that it is important that the learning process focus more on "primary concepts, not isolated facts." (Funderstading.com 1998) Constructivism seems to be the most widely accepted theory for learning, so it makes sense to use it as our education model.

In many cases it has been found that students learn mathematical process without learning the theory behind it. (Boyce 1995) This allows students to perform well on the tests without really mastering the concept. It has been brought up in several articles that mathematical learning should focus more on the concepts that are behind the equations. Matthews states that learning should focus "on conceptual understanding that uses a variety of intuitive, graphical, and numerical approaches" (Matthews 1996) to help students learn the concepts behind the equations. Matthew's idea is similar to that of Davis who talks about the rule of three for teaching. The Rule of Three states that students will have a better understanding of the subject if they "look at concepts from each of the numerical, algebraic, and graphical perspectives". (Davis 1994) Davis also pointed out that "interpreting results are important". (Davis 1994) It does not benefit a student to arrive at a solution if he or she does not know what the solution means.

Our particular area of interest in mathematics is differential equations and how it is taught. Hubbard, Boyce & Ecker, and Matthews all agree that mathematics should be taught through a variety of numerical, graphical, algebraic, and conceptual perspectives. It follows that differential equations should be taught similarly. Our experience at WPI has shown us that the numerical and algebraic perspectives are primarily dealt with

during lectures. In contrast the lab periods primarily deal with the graphical and conceptual perspectives of learning.

2.2 Teaching Mathematics with Aid of Computer Software

In differential equations it is important that students have visual confirmation to further their understanding of the behavior of the solutions. (Hubbard 1994) Students get this visual confirmation from graphs and plots of solutions, which in many cases are difficult to obtain by hand. Computers provide ways for students to get this visual confirmation through the use of mathematical software. (Ferrini-Mundy & Graham1991) Mathematical Software provides an easy and intuitive way to investigate the behavior of solutions and to model problems more efficiently.

In today's classrooms computer software is a big part of the students' educational experience, especially in classes where there are complex mathematical calculations involved. The use of software helps to abstract the method behind the calculation, and lets students concentrate on the concept. "Computer based simulations can expand an undergraduate mathematics instructor's opportunity to explore high-level mathematical concepts in an applied environment" (Salvit *et al*, 2002) without worry about the mathematical part of the particular algorithm used. But before we move any further let us define the computer based learning as done by Salisbury (1971):

> …A man-machine interaction in which the teaching function is accomplished by a computer system without intervention by a human instructor. Both training material and instructional logic are stored in computer memory.

This is the basic definition for any computer based learning, whether it is entirely through the World Wide Web, or just as a part of a laboratory assignment of a more traditional class.

There is another term that we must deal with, and that is computer-aided learning. Unlike computer-based learning, computer-aided learning is defined as "the use of (personal) computers for education and training" (HyperDictionary.com, 2003). The difference seems to be intuitive here; computer-based learning is entirely based on the use of computers. Course materials in this case are called courseware (HyperDictionary.com, 2003). Computer-aided learning relies on a more traditional class structure to teach new materials, some parts of which are taught with the aid of the computer. In the case of DElab, in WPI's context, we are dealing more with computer-aided learning. But the important question remains: how effective is computer-aided and computer-based learning and what can be done to make them more effective?

Numerous studies have been done on the effectiveness of computer-based and computer-aided learning. Studies vary from using simple algebra software for first year undergraduate students (Pierce *et al,* 2001) to a more complicated understanding of differential equations through contexts (Salvit *et al*, 2002). In one study conducted by California State University a class of 33 students has been broken into two groups: one group was taught statistics in a traditional method, the other group used an internet based approach. The quantitative end result "demonstrated the virtual class scored an average of 20% higher than the traditional class on both examinations" (Schutte). The qualitative side showed that the virtual class students had significantly higher perceived peer contact, and time spent on class work. They also demonstrated a greater understanding of the

material taught in class. This outcome certainly demonstrates that use of software and computers in teaching is indeed effective. Tapscott (1998) made another assertion on the basis of Schutte's study as proof. He said that "CAI (computer aided instruction) programs can improve learning performance by one third."

Ferrini-Mundy and Graham state that "Mathematical Software is growing and will be the way math is taught in the future." (Ferrini-Mundy & Graham 1991) Blanchard agrees with this claim by stating that "computers will play an ever increasingly important role in the learning of differential equations." (Blanchard 1994) The goal of using computers to teach mathematics is to free the students from repetitive symbol manipulation and to provide them with an understanding. This means that students will be able to focus on the process and what the solution means rather than the tedious processes that lead up to it. (Boyce & Ecker 1995) For computer based learning to be completely effective the students must be able to use the mathematical software efficiently. If students are unable to use the software because it is too difficult to understand or the software is used incorrectly, this may become a hindrance to learning.

Other studies showed that providing the student with verbal feedback could increase the usability. The study done by Gwo-Jen Hwang proposed that "a conceptual map model, which provides learning suggestions by analyzing the subject materials and test results" (Hwang, 2001) would improve the learning performance. A conceptual map is a form of a diagram with connections drawn between major ideas. In the context of this study major ideas had common problems associated with it, looking at those problems suggested which idea or topic a student is struggling with. An experiment based on this study revealed that "students who received the learning guidance generated by the

tutoring system, made significant progress compared with the control group." This shows us another way to make a help system more effective, by running some sort of analysis process in the background that checks for students' weak and strong points. This idea is worth pursuing and further investigation, checking for any constraints that MATLAB's environment could pose for development of the help system.

2.3 Help File Components

As mentioned before, the design of the interface of the program can influence the learning experience in different ways. The design problems are addressed by the Human Computer Interaction specialists, who had developed several specific guidelines, based on the research in cognitive psychology. Research and theory in cognitive psychology deal with the mental processes involved in perception, attention, memory, problem solving, conceptual information, and language. All of these mental processes are obviously relevant to learning and using computer programs and hence the design of help systems. Cognitive psychology also concerns itself with the subject of individual differences which is another aspect of obvious relevance to the design of help systems.

However, there are only a small number of principles that have emerged as generally applicable across all cognitive activities. Here are some of them, as listed in Kearsley and Norman/Draper books:

- The number of items that a person can attend to or remember for a short time is limited to about seven things (the concept of "chunking"). This means that any sequence or screen that requires the user to remember or attend more than five items simultaneously is likely to need help.

- When recognizing or learning something, people tend to generalize from what they already know ("stimulus generalization" concept). Which, in fact leads to the conclusion that if a program that works significantly differently from the way a task is done manually or the way other programs work is likely to need help.
- When understanding language, meaning (semantics) generally prevails over the exact form (syntax) of the language, which means that the displays or commands in which the exact format of the information is more important than the content are likely to need help.

To lessen the amount of help needed, one could follow these guidelines when designing the program in the first place, but if this sort of user-centered design is not possible, the developer must make sure that these issues are addressed.

As mentioned above, individual differences are important to the way people learn and use the new software. A number of experiments were conducted by the HCI specialists to determine the impact of different factors, called dimensions, on the usability of programs. These include gender, age, education, ethnic/cultural background, aptitudes, learning styles and personality traits, such as a person being and introvert or an extravert. For example, an experiment conducted by Yoder investigated the relationship between an individual's cognitive style and their performance on two tasks using the IBM VS/SP and VM/CMS operating systems with the help of different formats of help messages. Completion time, number of help requests, and errors made were measured as well as the participant's preferences for the different help formats. As the result, it was shown that almost all of the extraverts preferred graphic formats and most categories of introverts preferred explanatory formats. Perhaps of more importance, was the finding that most

participants performed better with formats based upon examples than the help messages involving abstract explanations. To accommodate these differences, the user should be allowed to customize interface parameters including the help system.

There are questions that arise from this information, and which are to be answered in order to create a successful system. Should the help be static or dynamic, how many levels of help should the system have, and how different they should be? What kind of access structures would be used, should the help be user or system initiated. What format should the messages take on the screen, and how extensive should they be? These are highly dependant on further research, but there are some things that we know in advance.

Taking the HCI guidelines into account, four different methods of creating a user initiated help system were created. The first type of system are those that answer the question "what is?", but require knowledge of a particular terminology for accessing the information they contain, such as the "man" command in UNIX. The second type is more flexible about the specific language. For example, synonym-based keyword facilities or natural language interfaces. Third are the "task-based" systems that provide information according to users' tasks and goals, rather than in terms of system boundaries or modules. And the last type of systems are the ones that bypass the need for the formal "help" by making the objects in the user's task environment "visible". One of the things that we should think about is which of those types to use when re-working the help system. In order to find the type to use, we should see which of the help systems currently available are successful.

One of the ways to see if the help system is working is to look at the feedback of its users. In the case of mathematics software the people we should consider are either

mathematics teachers or professors of practice, who are doing computations on everyday basis. This, however, does not mean that it will be necessarily easily usable by the main audience we are targeting, the students, since their approach to solving problems might be different from that of their professors. This issue was addressed when we took into account the individual difference workarounds adopted from the further HCI research.

There are reviews of the different software packages, written by the mathematics professors available from the The Maths, Stats & OR Network website, hosted by the University of Birmingham. From the reviews of the math programs on the website, such as *MATLAB 6 and Student version of MATLAB, Review of Mathematica 4.1, The Maple Computer Algebra System, Review of MathCAD 2001* and *Review of Mathematica 3.5,* we can see that there are certain components that must be present in the help system in order to make it successful. These include, but are not limited to: tutorials, detailed function references and descriptions, examples of both function usage and worksheets, and even training videos. These all tie in with the information drawn from the Yoder's experiment. Another important part mentioned is the layout and the ease of accessing the needed help system. MATLAB's Help Browser does the best job of this, providing the user with a search facility, index and table of contents. The examples are also compiled into an index for easy access. Programs such as Maple allow the user to invoke help directly from the problem spot and it either opens the help file, or goes straight to the on-line help, which is a significant part of all of the abovementioned packages. This part of the help system basically provides the full version of the Reference Manuals, which is more user friendly than searching through pages and pages of text. This is, by the way, one of the main reasons why the help systems were designed in the first place, to improve

efficiency. Having examples that can be executed directly from the help file is another valuable way to help the user. Maple and MATLAB both do this, and that is an another aspect that puts them on top of the useful help systems, as pointed out by Christian Beardah of Nottingham Trent University.

This short glance of the reviews suggests that the method of designing a help system taken by most of the developers of the mathematical software follows the third type of systems, but this decision is, of course only preliminary.

2.4 Other Considerations

In engineering practice, a good engineer must always look at unexpected implications that a design might pose (Vaz, 2003). One such implication in our case of DElab would be students not applying their knowledge of material learned in class, but rather relying blindly on the software to provide the answers, which consequentially carry no meaning to the student, because the topic is not even understood. If such a tendency develops it will imply that the help system for DElab is sufficient enough to provide help for users in order for them to use the software. In addition to that, the ability to verify one's answers is a very powerful tool in doing homework or preparing for exams. Answer verification provides students with the instant feedback of their performance. There is a good example from the past – calculators. "The purpose of allowing calculator use was to improve attitudes and increase student motivation to learn mathematics by increasing its relevance through greater use of applied examples. Very quickly the role of the calculator in the mathematics classroom expanded as teachers found ways in which it could be used not only to do calculations but also to assist students as they learned the

mathematics" (Pierce *et al,* 2001). More was said on the issue of calculators by Etlinger in 1974:

> Perhaps the most extreme view is that of the calculator as a purely functional classroom device...According to this view, the calculator will allow us to perform calculations much more easily, and will save us the trouble of learning the older, more tedious methods, much as the ballpoint pen saves us the inconvenience of inkwells and blotters...The pure-pedagogical point of view goes basically as follows: the calculator must not be used to replace learning, but rather to facilitate learning. Children must still learn their facts and their algorithms together with the more abstract concepts and ideas of mathematics. (p. 43-44)

This is an answer to the posed ethical problem. DElab or any other mathematical software for that matter, are meant to take away the tediousness of some tasks associated with doing real-world applied examples. This outcome defines the role, which DElab should play in the curriculum of a classroom. It should be a tool that aids students in working with difficult problems, which are not easily done on paper. It cannot replace the teacher, but it can help the teacher. A good help system will also help the teacher to explain the proper way to use the software, therefore the students will be more concerned with learning the topic rather than learning to use a particular function.

2.5 Learning Mathematics and the Help System

In order for us to create the best help system possible we have to take into account each of the topics brought forward in the previous sections. The use of a concept map allowed us to make connections between our research and the help system we are trying to create. This section will detail the key areas that this chapter has found to be critical in the design of a help system in an attempt to provide a rough framework, which to base the methodology chapter.

Without any research it can be said that content is an obvious issue that must be considered when designing a help system. However in our case content to some degree emulates a teacher in the sense that it presents new information that a user is trying to learn. For this purpose we consider the Constructivism view on learning. (Funderstading.com 1998) This idea takes on a shape of learning by building blocks, where each idea that is being presented builds on top of the previous one. It makes sense to use this theory because it is one of the most widely used theories in mathematical education today. (Selden & Selden, 1993)

Organization of content is another serious issue faced by any help system or documentation designer. (Kearsley, 1988) Virtually all help systems utilize a tree structure with different sublevels representing subtopics of a particular topic. This is a very logical way to organize and present information, but the depth of each level must be considered.

Rudolf vom Hofe (2001) in his study on effectiveness of different aspects of the software mentioned that dynamic real-time updated objects are preferred over static objects. While this was said in reference to the graphs updated in real-time to reflect changes of variables, this concept can be extended help object, which are being updated in real-time. What is better for the DElab help system, Extensive use of dynamic objects or more static objects? This question will have to be answered before our team can design a new help system.

Sometimes accessing or getting to the help file can be a problem. Access point can be either buried deep down somewhere in the menus, in which case the user will struggle in finding it. Or the access point can be of system-initiated nature, where the

help pops up whenever the software determines that there is a need for it. (Kearsley, 1988) Some sort of a tradeoff between the two extremes must be worked out and decided upon. The access should be in the obvious enough location, however it must not interfere with the users' work.

A review of help systems for different software packages as well as Yoder's experiment showed that topic access within the help system is an important issue. There are a number of ways in which the user can look up the help for the needed procedure or a function. There can be a list of topics, or there might be a search function, or maybe even a menu driven help. (Brockman, 1986) Each case must be looked at and considered because each software package takes a different approach. On the other hand, having access to all three might be a good idea, but HCI generally advises against it.

Human-Computer Interaction is a primary field of study that deals with interface related problems. Whenever something new is being written or developed, user needs must be one of the primary guiding criteria for the interface. (Norman *et al*, 1986) Screen formatting and layout cannot be dismissed as a simple eye-candy for the user. Things such as the selection of fonts, which look minor at first glace, have big implications. For example research determined that fonts which include serif are generally read faster than those without them. (Brockman, 1986) Following the best practices of set forth by HCI principles will ensure that new help file conforms to today's standards.

Each of the areas mentioned above provides a framework for the help system. For our help system to be successful we needed to find information to fill in the framework provided by this chapter. We needed to find out what content components

should be provided, what HCI principles to use, and what should be done for general organization and layout. The next chapter, Methodology, will outline the procedures, which our team will follow to obtain the information needed to complete this framework.

3. Methodology

The research part of our project brought up a number of interesting points and considerations for us to look at. However literature on its own is not a sufficient source of information, as it might provide only one side of the story, other sources must be considered in order to draw a full picture. Two more sources were selected: professor interviews and software review. A list of research questions was also selected based on the literature review. Our goal was to answer the three following questions: what content should the help system provide? How should the help system be organized? And what should the help system have a for a user interface?

3.1 What Content Should the Help System Provide?

Our objective for this stage was to find out what content would be needed for our help system. The content is essentially all the information that is contained in the help system. The content includes areas such as the explanations of each function, proper syntax, and any needed conceptual information. The conceptual information includes information on the actual course material. It does not need to be in depth but it should help the student understand what the software is doing. We obtained our content primarily from a review of the differential equations course and by interviewing differential equations professors.

It was thought by the team that a review of the differential equations material was necessary for the project. This way we would be able to understand what DElab is utilized for, allowing us to provide the proper content in the DElab help system. It would

have been done through analyzing the text and the notes we had from differential equations class. From that analysis a list of the main areas and key topics would have been created. This list would give us an outline format for the differential equations course. However, when we started reviewing differential equations, it became clear that we would require much more time for this part than originally was planned. Even if we would have succeeded in reviewing the topics, they would still not be at a level of the experienced staff that teaches the course. We felt that the new help system should be written by someone with the experience of teaching differential equations.

If we would have done the review, and come up with the suggested outline, we would have had to meet with the differential equations professors so that they can evaluate our outline and also help answer any questions we have on the material. Originally we thought that having the completed outline would help us learn how deep certain areas of our outline are covered in DElab and which areas are not covered at all. This would yield a more compact list of areas on which to focus. However, with all the changes made to the original plan, we took the reverse approach, gathering the information that is only needed for the functions that are implemented in DElab and adding it to the help system.

The main information sources for the project were the meetings with the professors which helped us gather a list of topics they suggested to include for each function. These meetings helped us create the basic outline and content of the help system and provided us with guidelines as to which parts should be included. Areas such as proper syntaxes are covered more in the software review. Using this approach

provides our help system with excellent content which allows us to provide the best help possible.

3.2 Conducting Interviews

In order to get the information we wanted from the interviews, an interview protocol was created. We chose to conduct open-ended interviews, which have predetermined questions so that each interview will touch on the same areas. This type of interview provides some focus without putting limits on what the interviewee can say. From the responses we received we were able to locate key areas that we need to focus on. (MacNamara 1999)

There are two major types of questions, closed response and open ended questions. Since the interviews that were conducted are on a smaller scale, open ended questions were more appropriate. An open ended question is a question that allows the interviewee to answer in any way they deem appropriate. These are a valuable interview tool. When used in the proper environment open ended questions often provide the interviewers with more information than initially expected. This extra information could have been relevant to what we asked, bringing up points we may not have thought of previously. On the other hand some of the information was not applicable for our research and in this case we had to adjust our questioning depending on the information given. (MacNamara 1999) During the interviews conducted during this stage, professors had pointed out a few things that were quite useful for our subsequent work. We will have more information on that later in Chapter 4, where the summaries of the interviews are provided.

For our questions to be effective we had to have the proper environment for interviewing. A good interviewing environment is a place where the interviewee feels comfortable. The interview environment should also be free of any distractions. Since the majority of our interviews with professors were at their office, it was a suitable interview environment. (MacNamara 1999)

The way the interviews were conducted was to have one person ask the questions, while the two other group members took notes. We have scrapped the idea of using a tape recorder to record the conversation which would allow all three members to discuss ideas with the interviewee without having to stop to take notes. However, the interviews went smoothly enough even without the tape recorder. It was also important for us to remain neutral during the survey, since a strong reaction to something the interviewee says could have caused him or her to answer the following questions differently. During the survey the interviewees sometimes would talk for too long or lead the interview off its intended purpose. It was up to us to keep the interview moving the way we wanted it to. Handling these parts of the interview correctly was important for our success. (MacNamara 1999)

After each interview we had to complete any note taking that we had done. Taking notes quickly sometimes leads to abbreviations or summaries that need clarification. This clarification was provided right after the interview while the information was still fresh. We also discussed how the main points fit in with the way we originally thought it should be done. Some of the interview information that we have received contradicted our initial plans; these contradictions were taken into account, as shown later in the document. (MacNamara 1999)

3.2.1 How Should the Help System be Organized?

One of the major decisions that were made by our team in the implementation of a new help system was the content organization. The primary goal was to find more information on the two of the following key areas. Should the help system be static or dynamic? And how many levels do we need to provide help so that it is least confusing. Finding additional information about these areas required software reviews and professor interviews.

The main objective of the software review in this section was to find answers to some questions concerning these key areas. The outcome of the review showed us what some of the leading mathematical software packages integrated into their help system. It also provided us with information about how deep other software's help systems go in regard to the topic. Some feedback on which system is better: static or dynamic was also received. Knowing the list of features and the success level of a certain software package helped us see which features add and which subtract from the usefulness of the help system.

The key in this stage of research was to be consistent throughout the review. This is why before we started the review a list of specific criteria that are the most important to the user of DElab, was created. The criteria included the extensiveness of examples and other supplemental information present, usability to the user, layout, and ease of use in correspondence to the human-computer interaction standards. The latter included specific guidelines defined for the access structure standards, system/user help initiation ratio, and graphical interface usability. Each of these criteria was assigned a weight based on its

importance to the user; a score rubric was also developed. The score rubric describes how each criterion will be evaluated, and gives point assignment to each evaluation outcome. For example the evaluation of the aesthetics of the user interface criterion can be broken down into five different categories: very nice attractive layout, clean layout, neutral, somewhat cluttered, and visually unappealing. The desired outcome gets the most points, which would be five points in the example that was just mentioned. Once we had the criteria and score rubric finalized we moved on to the software review.

Our team had operated each mathematical software package, in particular the help system that comes with it. We then recorded the points that it received in the value matrix. The value matrix simply lists all the criteria on vertical axis, and all the products on horizontal axis. The intersection between product and criteria is where the points from the score rubric are recorded. When the point assignment was completed, points for each product were weighted according to the importance of the criteria and then they were summed up. The product that got the most points was the one that most closely resembles what the final help system of DElab must look like (in respect to user criteria). This knowledge provided us with some more data about the key areas that were stated earlier in this section.

Due to the fact that one source of information is often not enough to draw the full picture of the situation, it was necessary to use a second source of information to determine organization – professor interviews. The interviews conducted here were very similar to those conducted in the interview of differential equations professors, in the sense that they follow the same procedure. The only difference is the questions that were presented.

From the software review we have compiled a list of components and criteria for levels needed for the help system. It was have also concluded that static is the best approach for this help system. At the end of this stage of research we had sufficient information to decide on the number of levels that DElab should have to be least confusing to the user and on the static approach.

3.2.2 What Should the Help System Have for a User Interface?

Another important decision that we had to make before getting to the implementation of the system is the design of the interface. We had three main questions to answer here: What kind of access structure should we have for the system? Should the help be user or system initiated? What computer Interface factors, such as font, line spacing, and text color should we use to improve the usability of the system?

Access structure was the least important of the three because it basically asks what kind of commands the user should input in order to access the needed help file. Is it via an assigned button on a keyboard, like F1 in most systems? Is it accessed through clicking on a certain icon, for example, or bringing up a context menu of the command with which the help is required? The goal here was to discover the easiest way for a user to get to the information they need.

System initiated helps, such as Microsoft Office Assistant (Clippit) are sometimes useful, but for more advanced users they are either bothersome or screen cluttering. The same goes for the balloons in Windows XP that explain some items that show up in the taskbar panel. They can be quite frustrating for the user who already knows what is going on, and seeing numerous instances of them can clutter your workspace. The goal of the

research in this scope was to find ideal rates at which those help messages are initiated by the system. Also the user should be provided with an option of disabling them completely without loosing any functionality, thus rendering the help entirely user-initiated.

Using the correct human computer interface attributes provides the user with less eye strain, confusion, and a better understanding of the material, as pointed out in the Cognitive Psychology Research. There are numerous attributes which we must take into account, for example the right depth and breadth of the menu trees, so that the user is not baffled with the abundance of choices and is not lost in the complicated structure. Another example might be the amount of white space on the margins of the document being displayed. Too little or too much white space will significantly lower the reading speed of the user. These and other attributes and criteria affecting the user performance in understanding and navigating the help system were compiled before we started the software review process, in order to create a value matrix.

We took two approaches to getting the information that we needed to reach the above stated goals, one of which was the software review. We have followed the same structure and information recording techniques as in the part that was devoted to finding out the best way to implement the structure of the help system, namely score rubric and value matrix. The criteria and weighting were, of course, changed to better suit the research that was conducted in this part. By the end of this iteration we had found most of the answers that were needed to meet our goals. We also had a good idea of what our interface should look like. It should contain the best concepts from the reviewed software packages. To ease the task of software evaluation and matrix construction, we had

combined the software review parts into one, evaluating both the structure and user interface at the same time.

With this outline the interview of the Human Computer Interaction Professor David Brown was conducted. The interviewing process, yet again, followed the same procedures described in the content section of the Methodology. The interview gave us a second opinion on the criteria that we have defined and also pointed us in the direction of the ones that we have missed. We were hoping to get out of the interviews a clearer understanding of what needs to be implemented in order to make the human interface factors to be as beneficial as possible. Professor Brown helped us with this problem by sharing his view on the way he would design the interface for such a system, and also provided general guidelines as to what should be done with the current system.

At the end of this part of research we have defined the way the user will access the help system, how often and at which stages will the system initiate the help and the basic design and layout of the user interface for the system. With these goals, as well as ones defined in the previous chapters, we were ready for implementing our new help system.

4. Data and Analysis

4.1 Interviews

This section explains the data we obtained through expert opinion. We obtained the data through interviews of some of the professors at WPI. The professors teaching at WPI have many years of experience and can be considered experts in their fields. The information that we obtained from the professors gave us valuable insight on our project and brought up several ideas we had not previously thought of.

4.1.1 Differential Equations Professor Interviews

We first interviewed the differential equations professors. Since the differential equations classes are the primary users of DElab this was an appropriate place to start. Our literature review brought up some questions regarding conceptual information and what kind of background knowledge should be assumed by the help system. We wanted to ask each professor a list of these open ended questions so that we could get different views on certain areas that were of interest to us. The questions we asked the professors are attached in Appendix A of this report.

4.1.2 Professor Heinricher

Our first interview was with Professor Heinricher. Professor Heinricher provided us with lots of valuable information and brought up some ideas that we had not thought of. This section will explain the key points that Professor Heinricher's interview touched

on. The complete memorandum of this interview can be found in Appendix B of this report.

One of the areas that we wanted to ask Professor Heinricher about was the presence of conceptual information. The literature review had shown that the conceptual part of learning was often overlooked and we wanted to hear what someone who teaches differential equations thought about the subject. Professor Heinricher thought it is an excellent idea to include conceptual information in the help system. Often students can blindly follow instructions to obtain a graph but have no real understanding of what this graph means. Professor Heinricher agreed that an interpreting results section, to help students get a better understanding of the concepts, would be appropriate.

We also asked Professor Heinricher about the usefulness of a frequently made mistakes section. DElab like any mathematical software requires certain syntax and other features that can lead to errors or problems for the user. Professor Heinricher agreed that this would be valuable to users of the software.

One area that Professor Heinricher mentioned that we have not thought of previously was about some of the problems that students have with graphing. Graphing can often be difficult due to improper windowing. A student might not realize the part of the graph they are seeing is not giving them the whole picture. In many cases in differential equations it is difficult to find the correct windowing for the function being graphed. Professor Heinricher thought this is a very important area and should be addressed by any help system.

We asked Professor Heinricher about how familiar he was with the DElab Software. He was unaware that the software was functional and available to use. Part of

this was due to the complicated installation procedures required. It was also due to the fact that Professor Heinricher tried to use a previous version of DElab which was not compatible with the latest version of MATLAB.

The interview with Professor Heinricher went very well and we obtained much of the information that we were seeking. Professor Heinricher also provided us with some new ideas that were very helpful. Our next interview was with Professor Fehribach.

4.1.3 Professor Fehribach

Professor Fehribach had some similar ideas to Professor Heinricher but was more familiar with DElab, so he was able to give us more DElab specific comments. This section will explain the key points of the interview with Professor Fehribach. The complete memorandum of this interview can be found in Appendix C of this report.

Professor Fehribach was familiar with DElab and was able to make it work properly. He said he doubted a person with and average understanding of computers could install DElab and get it to work properly. This seems to be a problem with DElab, the fact that it is difficult to install. Our main focus is not the DElab program itself so we will not address this issue. This issue and others will be addressed later on as recommendations for other DElab IQP's.

We also wanted to know what kind of background knowledge the help system should assume from the user. This is important in providing the right level of help for the user. A program designed for professors would assume much more background knowledge than that of a program design for an undergraduate. Professor Fehribach said that we should assume a background in calculus and novice computer skills.

Unlike Professor Heinricher, Professor Fehribach did not think that the conceptual understanding was as big a problem as getting the program to do what the user wants. Professor Fehribach said that generally a student understands what he or she wants to do but the problem lies in getting the mathematical software to do it. Professor Fehribach agreed that conceptual information would be useful but the help system should be focused on the actual use of the software.

Professor Fehribach agreed that graphing is a problem with mathematical software. He thought that having a part of the help system that dealt with graphing would be good idea.

When asked about help systems and how they should be structured he thought that Maple had an excellent help system. It allows the user to run functions directly from the help or copy and paste into the main window. He also said it would be best to avoid a Microsoft style help system. Professor Fehribach was displeased with Microsoft based help systems saying that they are generally very little help and "pop" up windows that are unasked for.

The interview with Professor Fehribach gave us a different perspective than the one with Professor Heinricher. Professor Fehribach also gave us some useful information on what is useful to the users and what can be annoying or frustrating to them. The next section will bring all the main points from the interviews together to show how they relate to the overall help system.

4.1.4 Analysis of Differential Equations Professor Interviews

From the interviews with the differential equations professors and the literature review we concluded that there should be four content components associated with each tool or function in DElab. These components are Background Information, Interpreting Results, Frequently Made Errors, and Technical Help. By splitting up the help system into 4 different help components it should make the help system easier to navigate and more useful to the user.

Background Information is an important component of the new help system. The need for the Background Information section arises from the need to have an understanding of what each individual DElab function is actually doing. The Constructivism view on learning states that each new idea presented builds on top of the previous idea therefore the end-user needs to have the basic informational blocks laid out. (Section 2.1) Providing a simple explanation of each function and perhaps some example graphs will help the user understand what the program is doing and provide the user with a better understanding of how to use the function.

The Interpreting Results section comes from the need for the user to understand the answer that a DElab function provides. The Literature Review section 2.1 states that arriving at a solution is of little use if it is not understood. Helping the user to understand what the solution means is important and will help the student learn. In DElab most solutions are in the form of graphs. Helping users read the graphs they create will allow for better use of the software. Professor Heinricher agrees with this saying that users often have trouble graphing because they do not understand what parts of the graph hold

the solution. The Interpreting Results section will help users greatly with understanding the graphing process and what the solution represents.

The Frequently Made Errors section came directly from professor interviews. While interviewing Professor Heinricher he mentioned that a Frequently Made Errors section would be helpful especially in a program like DElab. We then presented this idea to Professor Fehribach. He agreed that a Frequently Made Errors section would be very helpful. Generally when using software for the first time everyone tends to make the same general errors. This content component would go over the most common errors and provide help for them.

The final content component is the Technical Help. This is the part of the help system that explains how to use the software. Every help system has this component in some form or another, this section is crucial to the effective use of the software. Our help is no different in this respect and will have all the necessary information needed to get DElab to work properly.

There are some other considerations that need to be taken into account as well. It may be necessary to have some sort of text file with the DElab software that explains how to install it. Both professors had difficulty getting DElab installed and working. This file however will not be contained in the help system itself; such a file should be included in the zip file that contains the DElab software.

These four content components should provide a complete help system unlike any help system before it. These four components remain, for the most part, unchanged through the rest of the Data and Analysis chapter. The next section relates to the

interview with Professor Brown. Professor Brown is the main Human Computer Interaction (HCI) professor at WPI.

4.1.5 Professor Brown

This section goes over our interview with Professor Brown. Professor Brown is the main Human Computer Interaction professor (HCI) at WPI. Our interview with Professor Brown was extremely valuable. We asked Professor Brown some questions we had for him (Appendix D) and then we went through the current DElab Help System with him. This gave us valuable insight into what an expert thought of the current help system design and what could be done to improve upon it.

The first thing that we asked Professor Brown was if he had any prior knowledge or experience with DElab. This is important to determine if his opinion may be biased. Professor Brown had no experience of DElab or knowledge of its existence so his comments on DElab were his reactions upon seeing DElab for the first time.

We first inspected the content of the current help system because we expected that content was the major issue with the help system. Professor Brown quickly dismissed that assumption saying that plain text help, albeit old fashioned, was not a big problem. After reading some of the text in the old help system Professor Brown said it had appropriate information but the style of writing seemed inappropriate for this type of application. He also said the writing seemed to have poor transitions from expert level help to low level help. For example all of the text was clumped together in big chunks regardless of what level of help the text was providing. The text should be separated into high level and low level chunks to aid in understanding. Proper writing that clearly

expressed ideas is important in help systems. Professor Brown said a help system is not much good if you become more confused by the help than by the program itself.

The next thing Professor Brown noticed was that when clicking on some of the links he was often taken to somewhere he did not intend to go or in some cases nowhere at all because of a broken link. We found that many of the links to the textbook were broken and often contained many typos. Professor Brown also mentioned that some of the descriptions for links did not seem appropriate. When looking at the description it is expected that there would be one thing, but after clicking the link we found something else entirely different. The links in the new help system should be clear and take the user where they expect to go.

The next thing that Professor Brown mentioned while browsing the various pages in the help was that there did not seem to be any consistency from page to page. We found that there was very little formatting done to make each page consistent with the next. This creates confusion for the user and is not good help system design. Professor Brown suggested that since DElab is toolbox of MATLAB that its help should take on a similar form to that of the MATLAB help. Users of DElab would be familiar with MATLAB and thus having a similar help design would provide the users with a familiar form of help. This formatting would include things such as text structure, link summaries, search, indexing, and table of contents. All of these features are discussed later in the implementation chapter.

While looking at the help system for DElab Professor Brown was frequently confused by whether or not he was in the DElab help or the MATLAB help. He suggested

associating a window color for DElab to distinguish it from MATLAB. We picked a light green for this color similar to the color of the chairs found in Gordon Library.

Professor Brown had much to say about the layout of the help system and the DElab program itself. Many of his comments directly influenced the design of the help system itself. The main points have been covered but a complete memorandum of the interview can be found in Appendix D. The next section will bring together the main points of the interview with Professor Brown into a form that we can implement.

4.1.6 Analysis of HCI Principles

This section will provide an outline of the design components that we have obtained from literature review and our interview with Professor Brown. This section describes some of the basic structure components of our help system. These components will be dealt with more in the implementation chapter.

One area of the old help system that needs improvement is the text structure. The text in the old help system is often very long and unclear. Rewriting this text and breaking into smaller chunks will help the overall form of the help system. The help system should not have extensive amounts of reading in it. Professor Brown said short examples can be more effective than paragraphs of text. Having smaller chunks of text will make it easier to understand according to the cognitive research in Section 2.3. Having proper text structure and formatting will result in easier reading for those using the help system.

According to Professor Brown consistency is a very important factor when designing any program or software. Our help system is based on the MATLAB help so it

should be consistent with the MATLAB format. We should also develop a standard for each type of help page and then apply that standard to the entire help system. Having a system that is consistent provides a feeling of familiarity and helps in navigating the help system.

Our access structure will need to allow for easy access to the table of contents which contains the list of all topics. This will also need to be similar to MATLAB for the sake of consistency. The access structure provides a way to locate the current topic in relation to the rest of help system.

In many ways our help system will be very similar to the MATLAB help structure but with our content components. The next sections present our software evaluation and our evaluation of DElab itself. These evaluations will provide us with a look at how our help system compares to others and to provide us with ideas for implementation.

4.2 Software Evaluation

With a large number of different mathematical software packages available on the market, come many different ways of help system implementation. Since much of the software's success comes from its usability we decided to look at different help systems, for different programs, for examples of how things should or could be done. To choose which software example suits our needs the best we went with the method known as value analysis. This method is often used by engineers to narrow down the number of designs, to those that will suit customer requirements the best.

Our value matrix has twelve criteria, which are broken down into three categories: visual components, features, and functionality. These categories were derived from

research done by Microsoft Corporation. Breaking up all criteria into three separate categories will help us locate help systems which might be better at one category and not as good in another. Then we will be able to structure the design of our help system in such a way that it takes only the best features of other help systems.

Visual components deals primarily with visual aspects such as layout of a page, font selection, color and so on. Other information that is evaluated here concerns secondary and pop-up windows.

The Features category lists special functions which are present in different help systems. These include a search function, table of contents, index or a quick reference lookup function.

The last category is Functionality. It contains criteria which are related more to the content and access, or in other words deal with the flow of the help system, such as interactivity of the help system, or it being context sensitive. This section is also evaluating conceptual information, such as graphs and examples.

The next step in value analysis is to setup a score rubric which will determine how each criterion will be rated. The most desired option should get a higher number assigned to it. Most of the items in the value matrix were evaluated using a zero to five scale. Zero here means a complete failure or absence of a certain feature, while five means an absolute success.

For less obvious criteria such as fonts we tried to see how well they conform to the standard set forth by the same help system design research done by Microsoft; for example, the absence of a font would represent 0. Very small unreadable fonts would

correspond to 1, readable but incorrect font selection would mean 2, and so on all the way to the well readable size and correct font selection (fonts with serifs are desired).

Each criterion gets a weight assigned to it. There is no rule which says how much each one should get; the numbers are somewhat arbitrary relying only on common sense. In this particular case the weights were assigned based on our unbiased impression of importance of any particular feature, as well as the information that was obtained from the professor interviews.

The selection of the mathematical software used in the review was limited by the availability of such systems on computers in different WPI labs. The software packages that were readily available included Maple 8 and 9, MATLAB 6, and MathCAD 2001. However, we decided that this selection would not give us a broad enough variety, and downloaded the trial version of the Mathematica 5 package.

During the process of the data gathering, the value matrix was created and filled out. Each grade that the component has received was explained and compiled into a memo, which can be found in Appendix E of this report, along with the completed value matrix.

4.2.2 Analysis of the Value Matrix.

Below you will find some information about the help systems that were evaluated during the software review stage of our project. This information gave us an idea what to try to implement and avoid during the implementation stage.

Maple 8

The first software package that was evaluated was Maple 8. It does decent work in the functionality aspect of the review, providing good context sensitive help, and incorporating interactive functions into help. Developers of this software package were not, however, as successful providing users with conceptual information. Both functions and graphs present in the help system had no explanation of their significance or any kind of interpretation of the results.

Most of the items in the "features" section described in the preceding section were present in the Maple 8 software; however, the index and quick reference are not available to the user. The table of contents takes a different approach from the standard tree structure. Search is good overall; however it performs worse than the other search functions that were evaluated. Many associative links are present, pointing to the topics relevant to the subject, bringing this feature of Maple help to the top rated among the packages that were evaluated.

Visual components, however, is what breaks this package. It has a unique layout, which often gets in the way of quickly locating a topic. Poor font selection and bad placement of the help panes lower the score in this particular review section.

MATLAB 6

MATLAB does the best job of providing a user with conceptual information among all of the software packages evaluated. It provides a solid background for any mathematical process with its clever use of associative links and quick references. The help in MATLAB has all the features of Maple 8's context sensitive help. It also allows the user to access quick help in the actual application window. However, MATLAB lacks

any truly interactive help functions and does not allow users to run functions in the help window. If a user should need assistance with the actual usage of the mathematical functions, MATLAB includes screenshots which explain "How" to do something.

The help system for MATLAB scored maximum possible points on all of the features that should be present there. The quick reference does provide explanations to most of the terms needing them, however it does a slightly worse job here than MathCAD 2001.

The design of the help system for MATLAB is overall nice and clean. Font selection is poor, according to the HCI standards, but since this font is used throughout the entire system, we should be consistent with it, and use it as well. Overall, this style of help is the preferred choice for DElab, following the consistency requirements brought up by Professor Brown.

MathCAD 2001

This is the overall best package evaluated. It has all of the same features that MATLAB does, and is even better in some of them. One of the most interesting features of MathCAD help is the so-called "Quick Example", which brings out the example of the function that you need help with. This is a truly interactive help. It allows you to change variables and observe the changes in graphs and results. It also allows you to run functions from the help window.

MathCAD also improves upon the quick reference feature, providing the explanations and background materials on even the "lowest" items in the tree structure (e.g., what is an integer?). It also uses a very well selected font with serifs and follows the standard toolbar and pane setup and structure, which is the most familiar to the user.

The only criterion that this help system received a lower score at is conceptual information. Descriptions of help functions are not as detailed as MATLAB but still do a good job. MathCAD also lacks the explanations of solutions and information on interpreting graphs.

Maple 9

Maple 9 has almost the exact same set of results as its predecessor. On the other hand, there are several differences that are worth mentioning. Designers of the new help system had to make a few tradeoffs in their design. For example introduction of the index, which was missing in Maple 8, reduced the usefulness of the associative links, as a result, they are not as numerous. The table of contents that is used in Maple 9 looks a bit awkward without the standard "+" signs to signify expanded topics. However, it is still more familiar to users, since it now uses a tree structure. Most of the other features, which were left unchanged, are comparable to those of Maple 8.

Mathematica 5

This is a hand down the worst help system that we have reviewed. It practically copies the layout and structure of Maple 8 but lacks some features present in it. On the conceptual part, it lacks detailed descriptions of mathematical functions as well as usage help. It is even harder to find some relevant information to the topic you are trying to get help about because the associative links are almost nonexistent. The creators of the software assume that the user will find the information with the other means provided, however the index is awkwardly implemented and is hard to use. The only thing that

really shines in this help system is the fact that graphs inside the help system can be updated in real-time, very much like MathCAD.

Overall, MathCAD came out as the help software with the best functionality and is on par with MATLAB in other two sections. MathCAD provides decent conceptual information with many excellent interactive examples. The ideal help system design for DElab should have used some of the stronger points of MathCAD, but since we wanted it to be consistent with MATLAB, most of the features could not be implemented without further extensive development. Simply adapting MATLAB help was a good approach to creating help files for DElab. It is one of the top scoring help systems in our review, and adapting an existing system reduces development time.

4.3 DElab Evaluation

The original DElab help system is far from perfect. Compared to the software packages discussed above, there is a complete absence of any conceptual information related to the mathematical functions that are being performed by the toolbox. However, not all of the required content is missing. Examples that are present in the current help system should be left in the help system, to be consistent with the current differential equations textbooks edition. Some of the instructional parts of the old system could be integrated within the new design. It will be easier and more effective, however, to replace them with instructional pages that have a structure similar to that of MATLAB or MathCAD. These pages should be complete with the screenshots of the various windows used by the function and the description of each component that is present on the screen.

Another noticeable problem is the Human Computer Interaction features. Currently the help system is very far from the MATLAB help in this respect. As was pointed out by Professor Brown, there are several drawbacks that should be addressed to bring the system up to the current HCI standards.

There is no use of color whatsoever in the system, which is an important tool to draw users attention to particular parts of help, or indicate the current "location" of the user. Better understanding of the information provided by help requires the introduction of such elements as example graphs, tables, and screenshots. Easier navigation between topics should be implemented by linking the currently disabled index panel to the help files. This is crucial for better understanding of the information, as well as reducing the frustration that the user might experience while trying to find relevant information. These and other problems are described in more detail in the chapter concerning the interview with Professor Brown.

There are currently issues with the organization of the content as well. There is no hierarchy present or at least apparent in the system right now. This comes from the fact that the current help files have a flat structure, and the organization is somewhat arbitrary in this case. Some easy changes can be made by splitting the old DElab help files into smaller pieces, to make finding something specific easier for the user. Currently it is a problem finding information about a specific DElab function, since it may be bundled together with similar functions in a larger file. One of the most important tools to find something relevant to the displayed help, the reference or associative links, are missing or broken. This also could prove to be a challenge for a user trying to navigate the system. More relevant links should be introduced, and the broken ones should be fixed.

Since the organization is not clearly defined, the old system did not have a use for such things as table of contents, index, or search functions. To make the new hierarchically organized and complex system operate efficiently, these should be implemented. Table of contents will provide the users with an idea as to where they are currently in the help files. This will allow the user to perhaps find needed information straight from the tree alone. The index function is self-explanatory and should be always present where large amounts of data need to be processed. The search function is as important, since it allows the user to find the needed information quicker.

With this information known, we now have a good idea of how much work should be done on the new help system, how much can be reused, how much should be discarded, and how much should be implemented. This is discussed in more detail in the following section.

4.4 Overall Analysis

Below is a summary of changes that need to take place in order for DElab's help system to be consistent with modern standards. In fact, these modern standards, obtained from research and data analysis, define the new help system. To make reading a little bit easier, the list of changes was broken into three major types: items that need to be added, items that need to be changed, and items that need to be removed.

4.4.1 Items to be Added

Our research divided the entire structure of the help system into two major components: content (text and meaning behind the text explaining a particular software

function), and human-computer interaction components (visual aspects such as font type, size and color). Each of these components is equally important to the success of the whole system, because one appeals to extravert type people, those who prefer graphics and other types of visual aids, and another to introvert type people, those who prefer explanatory format, as discussed Yoder's experiment in section 2.2.

Interviews with Professor Heinricher and Professor Fehribach showed the need for new sections such as "Interpreting Results" and "Common Errors." According to those professors, students make a lot of common mistakes which can be easily solved if users are directed properly. On the other hand, if students do not know what they are looking for they are very likely to interpret the result that they obtain incorrectly. Therefore it is very important to give users a list of common errors that can occur, and supply them with solutions for those errors. It is also important to explain how to interpret results in order to avoid "blind" copying of answers. This idea of interpretation addresses an ethical issue posed by mathematical software packages which is discussed in section 2.4.

Another new section that was suggested at the interviews was more related to the differential equations course itself. Literature Review chapter talked about the need for the help system to be, at some extent, at the level of a teacher, where a software function would be explained from mathematical point of view rather than just mechanical "how-to" list. The proposed "Background Information" section draws the bridge between the differential equations course and the software itself. In a way it motivates the need for the requested software function by explaining where the function arrived from, and in what ways it is useful, all in mathematical terms. A good example would be Direction Fields

software function explained in more detail in chapter 4. The background information given for that function describes, with equations, how the slope lines are created. Therefore the user does not just blindly obtain the solution, but he or she knows where the solution is coming from.

The new sections explained in the previous paragraphs must be written from scratch. The text for them does not exist in the old help system; therefore nothing can be adapted or transformed.

A few words about the text itself, "Background Information" can be taken from the textbook, with only minor adaptations required. It is already written at the college level, specifically for students taking differential equations. Any other users who are not students should be sufficiently comfortable with the topic to understand language used in the textbook. The "Interpreting Results" section is trickier. It requires a thorough understanding of a differential equations material, and it should be written by someone with experience of teaching the course and having the knowledge of where to direct students' attention when solving a particular problem. The "Common Errors" section cannot be written right away, because not all errors are known at the time. It will take some time before the list can be accumulated. The process by which the list is actually expanded can be implemented in a number of ways. One convenient way is to use short report forms, which can be easily realized as a part of an HTML based help system. User entered information will be sent out via email to a predefined email address. With the growing database of common errors subsequent updates to the software should have updated help system files. The actual implementation of this idea is left for the next phase of the project.

Professor Brown in his interview emphasized consistency. Having identified it as a primary driving factor in HCI components, our team decided that the most appropriate interface for the new help system will look like MATLAB's help interface. There are a number of other advantages to it besides being consistent with MATLAB. MATLAB already provides an easy interface for implementation. It takes care of many help functions such as Search and Index, which were also identified as necessary, and will be explained later in this section. It also keeps everything within one window, which avoids clutter on the computer screen.

Consistency is also needed at a lower level for the pages themselves. As seen in many other mathematical software packages, and as identified by Professor Brown, all pages in the new help system must be subjected to a common formatting standard. Such factors as font type, text color, text size, image size, image placement, table formatting, etc must be kept consistent throughout the pages. Since there is no common format in DElab as of this moment, this formatting standard must be developed. The new standard proposed by our team is discussed in detail in the next chapter - Implementation.

Software evaluations showed the need for help functions (also identified as help features) such as table of contents, index and search. All three of these are located in the left panel in MATLAB's help. The current help system does not have support for those help functions; therefore they must be designed and developed from scratch.

4.4.2 Items to be Changed

From the data obtained at the interview it became apparent that each help page must be broken into subsections. A few of those subsections were already mentioned in the previous section. Taking into account the sections that were deemed to be necessary, our team arrived at the following structure for each function: Background Information, Technical Help, Interpreting Results, and Common Errors. This is new, compared to the old help system's approach to information organization, for each software function explained in the help system. Some software functions are already mentioned in the help system therefore not everything has to be done from scratch. The "Technical Help" section can be adapted from the current text. The section explains how to use a particular software function in a step-by-step manner. Nonetheless more details and screen shots should be added, to make it easier for graphically oriented people, who learn from graphics and pictures faster than from the abstract text.

The current DElab help system contains a number of examples, which are also referenced in the text book. These examples explain how to solve a problem using the DElab software. From Yoder's experiments (explained in the section 2.2) section our team concluded that examples must stay in the help system, because most people learn by examples. However, current examples cannot stay where they are in the organizational hierarchy of the help system. They must be moved into a separate section, where they can be easily located. Professor Brown suggested that it is preferable to break up the text into

smaller pieces, thus we propose to organize all examples by chapters. Also each example must have its own page, if possible with graphs and screenshots.

While evaluating the current DElab help system our team noticed that pages contain reference links which often are not related to the topic being explained on the page. Many times links were not even clickable. To fix this problem all reference links must be changed to point to right pages, which are conceptually relevant to the topic being looked up.

4.4.3 Items to be Removed

The list of items to be removed is quite short. Most things must be either added or changed from the existing help, however there are a few items which are not needed.

Non-working reference links mentioned in the previous section must be removed, in order to avoid confusion. As noted by Professor Brown it is a bad practice to have links that lead to nowhere.

Another aspect of the current help system that must be abandoned is the formatting style. Current formatting style is inconsistent, and hard to read. All text is grouped together, and there are no figures or screen shots whatsoever. Headings and subheadings do not stand out, and overall it is hard to locate an item on the page. All of these must be abandoned in favor of a new formatting that should be similar to MATLAB's formatting.

Some things mentioned in this section are not concretely specified, such as a font that should be used in the new help system. The selection will depend on the current

MATLAB formatting for reasons of consistency. Things that were not described in this section should be assumed to be the same as in MATLAB's help. MathWorks, the developers of MATLAB, have a number of years of experience behind their software and their help system. Therefore whatever they use in the help system must work for them. After all, the success of a software package is a good indicator of its help's usability.

5. Implementation

5.1 Implementation Plan

This chapter describes the original plan for the implementation of the new DElab help system. It is important that the plan is followed closely in order to create a help system that follows the HCI standards, as well as the writing and formatting standard derived from the research part described in the previous chapter.

The first item that needed to be addressed was the HCI components of the new help system. In order to do that, we have picked a formatting standard that was applied to each help page throughout the system. Most of these components were designed in order to keep the DElab documentation and help files consistent with those of MATLAB. This is especially true when it comes to picking fonts for text, heading and subheading of each page. The heading should, for example, be in Arial font of size 14, since it is the style and size of font used throughout the entire MATLAB system. This also means that such heading formatting should be used throughout our own system, no matter what page the user views. This type of standard also applies to other page elements, such as the font color (dark red) for the subheadings on the page. Text font, color, and size should also be taken from the overall formatting of the MATLAB help files, in order to be consistent.

Since each of the "chapters" in MATLAB'S help has a color associated with it, the DElab should have a color associated with it that is unique from the ones already used in the system. It is important to keep the same color throughout the help pages as a visual indicator to the users about their position in the help files. This will tell if the user is still viewing the DElab help, or some other part of MATLAB. It is especially useful when

having links that point outside of the DElab help system. This color scheme shows the users that they are no longer in the DElab help system.

The links themselves should also be placed in specific parts of the page, if they are not contained within the text itself. This will make the users more comfortable with the system, since they know where to look for the links on each page. This is better than having the user scroll through each differently formatted page trying to find the relevant link. This also makes it easier to implement changes, linking newly introduced software features to the pages they relate to, without having to look for all the different places where such associative links might be placed.

Another page formatting item that should be addressed is the margins, which should also be consistent with those of MATLAB, as well as on a page to page basis within the DElab help system itself. The margins play an important role in the way a user reads the page, and as such should not be either too small or too large. These extremes not only hinder the users' ability to read the text quickly, but also have an impact on the understanding of the subject. MATLAB developers have chosen the margins to be reasonably large, and we should follow their standard when it comes to this HCI component.

Graphics and table formatting is also important, as they also have an impact on the users' perception speed and understanding. Graphs should be big enough to show all of the necessary details needed, however they should not be too large, since they will obstruct the view and hinder reading (having to scroll down past the graph or image considerably slows down the reading speed). In the event that the users might need more

detail than the help provides, they can always run the necessary commands to reproduce the graph and get the necessary details from it.

The best choice for explanatory screenshot sizes, such as of those showing an example of function usage or options, is the size of the window containing the options itself. This way it is easier for the users to relate the situation on their screen to that described in the help file. This type of screenshot should ideally show the entire window, as to not confuse the user as to what part of the window they are looking at.

Tables should be formatted consistently with those found in the main MATLAB help files. Since the text formatting also corresponds to that of MATLAB, the tables should have the consistent proportions and attributes for this application. This might include the height of cells, border thickness and other items.

In order to navigate the new system effectively, some changes should be made to the navigating features that were described in the previous chapters, such as the Table of Contents, Index, and Search functions. Currently these functions do not work with the old help system, however they fit in well with the new help system. For example, the organizational tree that is located on left part of the MATLAB help system should expand and/or collapse in accordance to the page currently viewed by the user. This is one of the crucial requirements for navigation, since this tree gives the user the best idea about his location in the system. The index tool should also be changed so as to incorporate all of the items in the new help system. The user should be able to find the required help page just by typing in several keywords, if they are not familiar with the structure of the help system. The search function should also be able to find any important term introduced in the new or old help system.

In order to show all of the components that are required for a full-featured help system, a complete section should be implemented, using the guidelines above. This involves writing or rewriting the text, creating sample graphs and tables, as well as operational screenshots, and putting them all together into structured pages according to these standards. These pages should also be organized, searchable, and accessible through the three tools described before. It will serve as an example for future development.

5.2 Obstacles Encountered

At the initial stage of implementation, our team had encountered errors in the DElab software preventing us from finishing some tasks that we initially set out to accomplish. Some of the errors concerned the graphs. This was a problem for the technical part of the help system that explains the steps required to reach a particular solution, and the interpretative section could not be completed with the results not being displayed correctly.

As it was later explained to us by Professor Davis, these bugs were the side effects of the version incompatibility of the current DElab build and the MATLAB software in use. DElab was coded for earlier versions of MATLAB. When these compatibility issues are addressed, the help system will describe the correct function usage and help interpret the correct results. Some issues arose from the fact that the team was using a different version of the DElab than that used/developed by Professor Davis. These problems were evident in the lack of the certain menu options or availability of certain equations. These problems were not crucial for the part of the implementation that was completed, but should be corrected when the complete help system is implemented.

These obstacles hindered our progress in implementing as much of DElab help system as we wanted. We were still able to complete a good part of implementation to serve as a framework for further implementation. What we did implement is discussed in the next section.

5.3 Accomplishments

This section discusses the accomplishments of the implementation stage of our project. While there were a few obstacles mentioned in the previous section, overall the implementation took a good shape and serves well as an example of how entire help system should be implemented based on guidelines set forth by this report.

5.3.1 Direction Fields

Our team chose one software function to be fully implemented according to the guidelines discussed in the Analysis chapter. The software function chosen was "Direction Fields" located under the Graphical Tools menu. The reason for choosing this particular function was simple: conceptually a direction field is one of the easiest topics to understand. Therefore we could write more about it, with less conceptual errors.

In Chapter 4 we proposed a new help structure for each software function. According to this proposition the help topics were broken up into four sections: Background Information, Using the Direction Fields Window, Interpreting Results, and Frequently Made Errors. This part of the document will discuss each section.

Background information was not written by our team, instead we chose to use text supplied by the textbook "differential equations" written by P.W. Davis. The visual aspect of this section was organized in the same manner as discussed in Section 5.1. The result is depicted in Fig. 5.1.

The next section implemented contained the directions on how to use Direction Fields window. Text was written by the team as the team understands this window's functionality. Also, this section included a number of screenshots to make the understanding of a window's functionality easier. The final outcome is shown in Fig. 5.2.

The next section did not actually get written due to conceptual complications. The section called for the text that would explain and interpret results. Due to the nature of this text, it would require someone with the experience of teaching the differential equations material. In order to save time, and avoid errors this help screen was left empty, only the outline for this section was written. The outline consisted of four sections. These sections are listed below with explanations for each one.

- *Examples* - will either link to text examples or will contain the separate examples
- *How to do by hand* - this section will contain information on how to create a direction field plot by hand
- *Graph* - this section will contain a graph of a direction field
- *Interpreting the graph/solution* - this section will interpret the graph under bullet three and point out areas of interest. These areas may include information on solution lines and the stability of the model

To stay consistent with the rest of the help system, the same HCI standards were applied to it. The result is shown in Fig. 5.3.

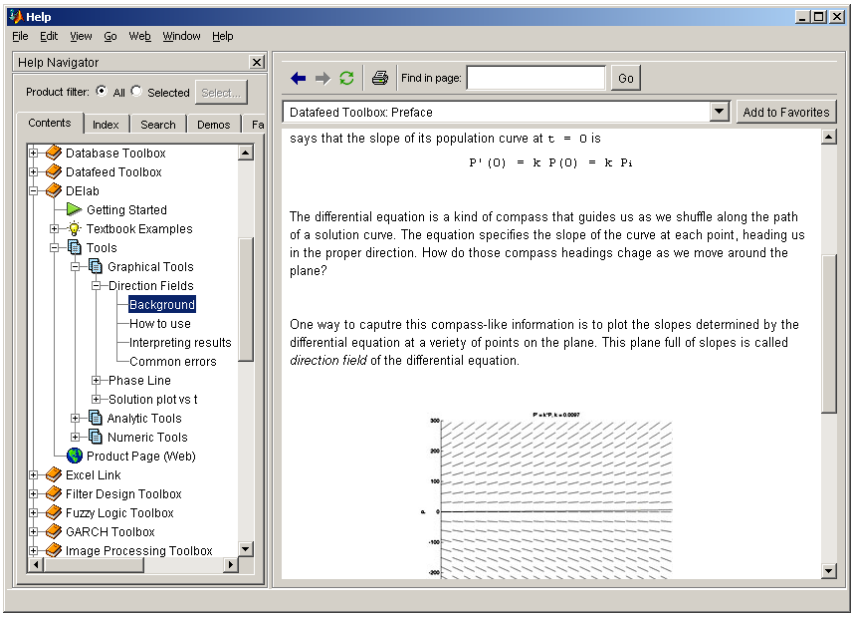

Figure 5.1: "Background Information" Window

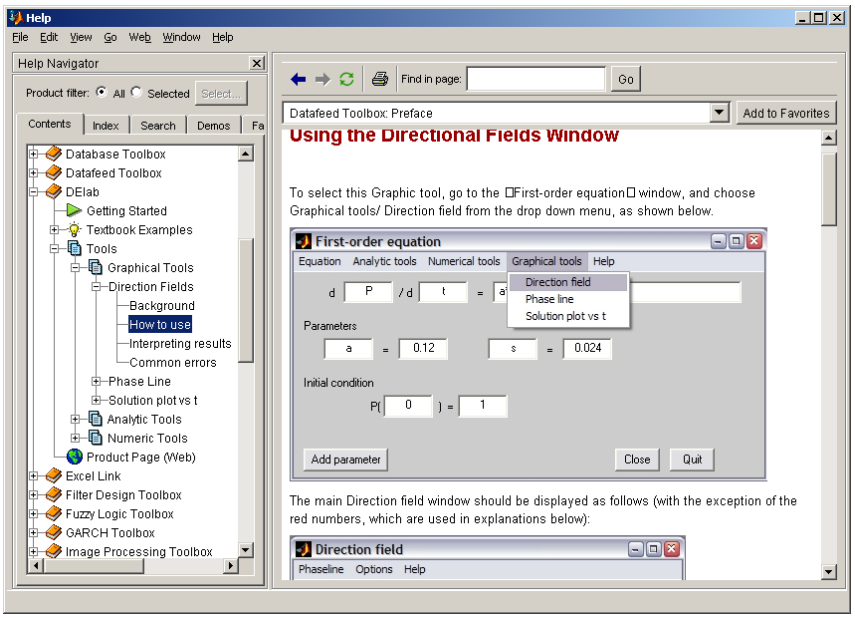

Figure 5.2: Using the "Direction Fields Window" Help Screen

Frequently Made Errors section was implemented in text to the best of our abilities. The nature of this information requires a long process of compilations, with commonly made mistakes slowly accumulating into a thorough summary. Due to our time restraints, our team had to limit the list to three items mentioned at the interview by

Professor Heinricher. All visual aspects of this page were brought to the standard used throughout the new help system, and can be seen in Fig. 5.4.

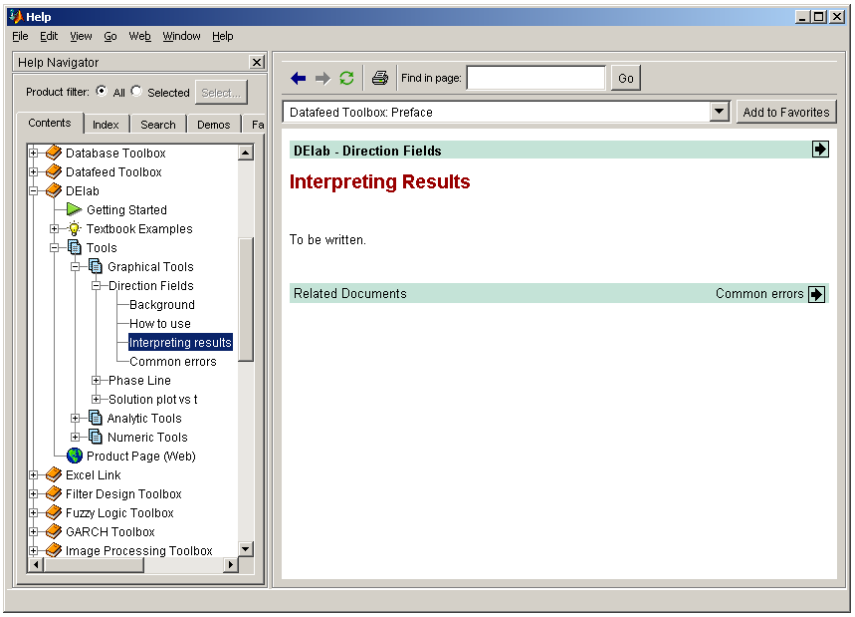

Figure 5.3: "Interpreting Results" Window

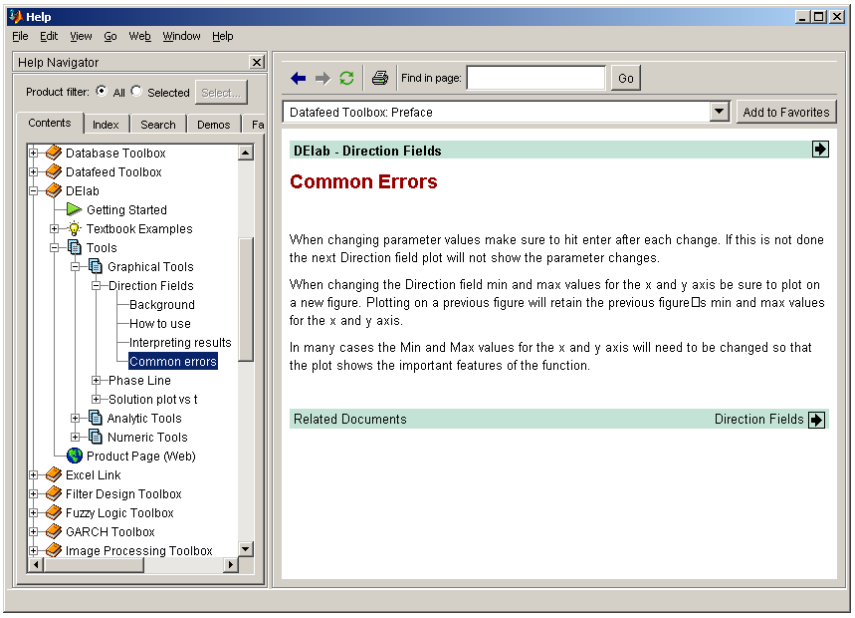

Figure 5.4: "Common Errors" Screen

5.3.2 HCI Principles

The Data and Analysis chapter showed the need for a unified formatting standard, also referred to as HCI formatting standards in this text. When this format was developed the next step was to implement it. This section will explain the actual implementation of those standards.

Since consistency was the major factor in all of our decision, the final format was very similar to MATLAB's format: same dark red color was used for section headings, same Arial font was used for general text, and same single spaced paragraphs separated by a single line empty line. For mathematical equations we went with the same type of mono-space font used in MATLAB's help system. The list of minor details is quite long therefore we included screen shots, shown in Figs. 5.1-5.4, of the new look as well as a digital medium containing all the source files.

Making the Table of Contents work was a major part of our implementation. All the technical specifications are described in Section 5.4. To the degree that the concepts extend, the entire table was broken down into two major sections, Textbook Examples, and Tools; see Fig. 5.5 for a screen shot. The Textbook Examples section was further broken down into chapters, and chapters were broken down into examples. Tools were separated into three major subsections such as Graphical Tools, Analytic Tools, and Numeric Tools. This organization was based on the flow of the user interface in DElab.

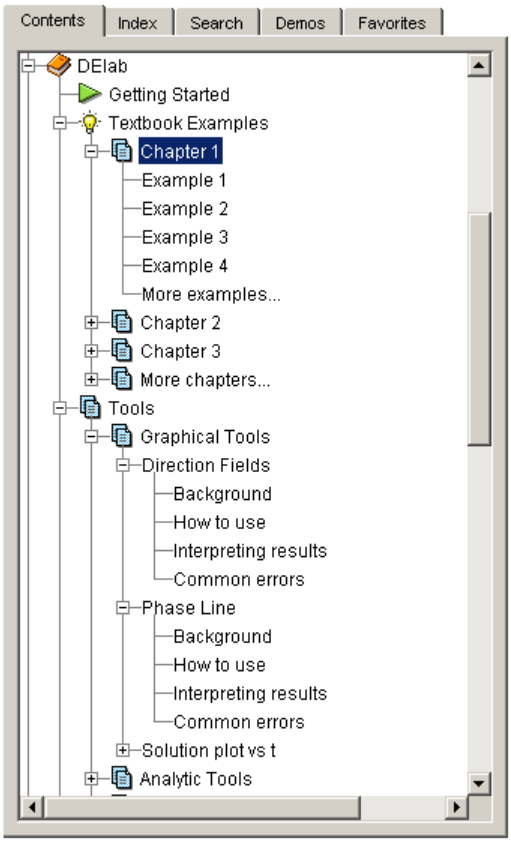

Figure 5.5: Table of Contents

The Textbook Examples section was organized by chapter for easy reference. Also, in hopes to keep it consistent with current references in the textbook, our team used the original example names such as "Figure 1.2 Example." All of the examples were formatted into a tabular form, with small descriptions on the right side of the table. The main motivation for providing the descriptions came from Professor Brown's interview. At an interview he said that including a small text that lets the user know what they will find inside the link is a good idea. This idea of a tabular form is shown in Fig. 5.6.

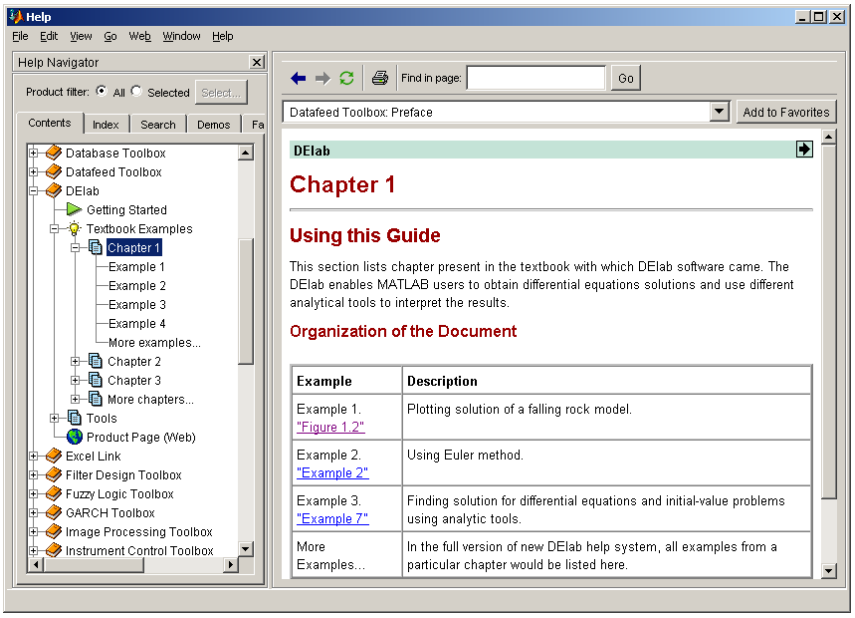

Figure 5.6: Chapter 1 Examples

Another important section that our team set out to do was an Index function. Index helps users locate information in a timely manner, and is very similar to an index present at the end of almost every book. MATLAB provides an interactive index function, where the lookup happens in real-time.

Our team implemented a short index list, with only four to six search terms, as a proof of concept. When the need arises, this index can be expanded to any number of terms. The technical information on how that can be accomplished is in Section 5.4. An example of a working Index function is shown in Fig. 5.7.

The last HCI component to be implemented was the Search function. MathWorks, the creators of the MATLAB software, stated that the creation of a search function under Windows operating system requires familiarity with Perl and Perl modules. More information of this topic can be found at the following internet location:

http://www.mathworks.com/support/solutions/data/1-18U6Q.html

5.4 Technical Specifications

The following section will serve as a quick guide for the average user on how to modify some of the more advanced features of the new DElab help system. It assumes the basic knowledge of HTML and XML, and by no means serves as a guide or a tutorial for the aforementioned technologies. To make it easier, this section has been broken into four parts: help structure (filenames, locations, etc.), overall layout, table of contents, and index.

5.4.1 Help Structure

MATLAB has a relatively simple help structure, which is very flexible in its implementation. There are two basic technologies at use, HTML and XML. HTML is used to display pages in a standard Internet Explorer engine. XML is used to organize content. Files for the help system can be located in any folder as long as the correct path to them is provided (either relative or absolute). The path to DElab's "Table of Contents" file is stored inside the master "Table of Contents" XML file, called "mwdoctoc_delab.xml" located in /help folder. There is no need to modify any MATLAB files due to the fact that MATLAB automatically recognizes the "mwdoctoc" prefix and includes needed structures in the master table of contents. The same sort of linking technique applies to the index file, which is linked to from the master index file called "mwdocindex_delab.xml." All HTML files are linked to from within the XML files.

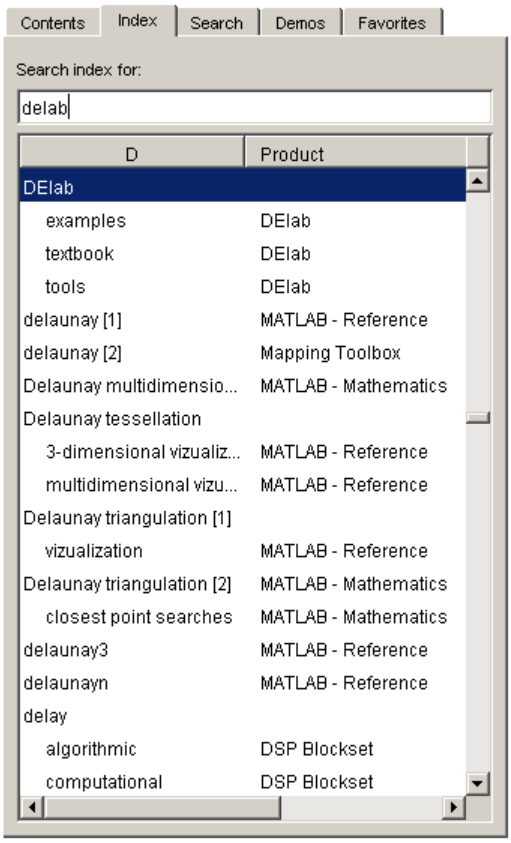

Figure 5.7: Index

5.4.2 Overall Layout

Currently DElab's new help system is located within one folder help\toolbox\delab. This folder contains all images and all HTML files. When the image is used in the HTML file the relative path is provided to assure portability to any other path. All files are linked to a single style sheet file, which defines such aspects of the page as heading color, heading font, font size and so on. If those items should ever be changed, there is only one place where they have to be modified. This file is called "docstyle1.css" It is a standard CSS file.

Filenames currently assigned to different sections can be changed at any time, as long as they are correctly updated in the organizational XML files (more will be said in

the next section). If new files are to be created, they should also be added to the XML file listing.

5.4.3 Table of Contents

Table of Contents is implemented by the means of XML technology. All content is written in a hierarchal manner, which is rendered by MATLAB into a clean tree, as seen in Fig. 5.5. Entries in this XML file look similar to this:

```
<tocitem target="preface.html" image="$toolbox/MATLAB/icons/greenarrowicon.gif">
Getting Started
</tocitem>
```

Target provides the name of an associated HTML file. Image provides the link to a small image that appears next to the topic title. The images do not have to be limited to those supplied by MATLAB. Text enclosed within “tocitem” tags corresponds to the text that users see in the Table of Contents.

All information is stored within the “delabtoc.xml” file.

5.4.4 Index

Index help function is achieved the same way “Table of Contents” are. The XML file contains the search terms along with the associated HTML files. Entries look similar to this:

```
<indexitem>DElab
    <indexitem target="tools.html">tools</indexitem>
    <indexitem target="textbook.html">examples</indexitem>
        <indexitem target="textbook.html">textbook</indexitem>
</indexitem>
```

The higher level item is the major topic, and the lower level items are specific topics within the major topic. A good example would be a topic of graphs broken down into direction fields, solution plots, phase lines, etc.

All index information is located inside the “delabindex.xml” file.

5.4.5 Search

Our team was unable to implement the search function as part of the MATLAB help system although some information was discovered on this issue. The list of terms associated with DElab help system is stored in the database file called “index.db.” This file can be generated using a Perl script distributed with MATLAB R13, and located in folder help\base\util\build_search_database.pl. However, there is a complication; Perl is not easily installed under Windows. To get around this issue, it is suggested to generate the database file under Linux or UNIX and then simply copy it over to Windows for distribution.

6. Conclusions and Recommendations

This section summarizes results and accomplishments. With this first phase of the larger help system project complete, and the framework finalized, what are the next steps? What can one do after he or she finishes reading this report? What needs to be done in order to have a complete and functional help system? The final chapter of this project will try to answer all of these and other questions.

6.1 Further Steps

After reading Chapter 5, Implementation, one will get a sense of where the current project stands. The research part of the project has been completed. The framework for the new help system was established, based on literature, interviews and software reviews. As a result, new help system is up to modern standards of help design. It includes such interactive features as expandable table of contents, which automatically shows the relative location to other documents, real-time lookup index, and search functionality. The new help system is also very closely integrated with the current MATLAB help system, which makes it more consistent with the rest of the software packages that come with MATLAB.

The entire help system is by no means fully complete. There are a few items that must be taken care of before the system can be released to the public. These items vary from very minor changes to interface text, to major changes such as new text for main sections of the help. Below is the bulleted list of our recommendations:

- One section was implemented as a proof of concept, but there are many other sections which are still lacking both technical and conceptual content. Those sections must be completed with the information needed. Our team recommends someone with the experience of teaching the differential equations course to write the text for the unimplemented sections.
- One of the major obstacles encountered during the implementation stage of the project was the failure of DElab software to operate properly. The nature of the errors was described in the previous chapter. And while they are not crucial, they interfere with a normal usability of the program, and if it is possible, they should be fixed.
- There are some other flaws detected in the software itself. Professor Brown while evaluating the current DElab help system found numerous HCI rules violations in the software's interface. These were things such as poorly placed buttons, unclear menus, and other simple errors that could be easily corrected. This list is part of an interview transcript, which is located in Appendix D.
- Chapter 4, Data and Analysis, introduced the notion of a Common Errors section. It briefly mentioned that this section cannot be compiled in a short period of time. Instead some sort of a feedback scheme can be developed to accelerate the accumulation of the common errors. The design and the development of this feedback scheme are left entirely up to the team continuing this project's effort. However we recommend having a separate HTML page, which would serve as a communication tool for all feedbacks. The form can include a field for person's name, email address (in case they must be contacted to obtain more information),

and comments box, where the user can describe the nature of an error. This sort of scheme can also be used as a bug-tracking tool. Another recommended way of obtaining more information on commonly made mistakes is a study group. A group of students can be placed in the controlled environment, and asked to do some problems with the aid of DElab. Mistakes made, and their solution, can be recorded and used as a basis for the Common Errors section.

- One last idea that was only briefly touched upon in the Professor Heinricher's interview, and later on reiterated by Professor Fehribach, revolves around the difficulty of installing and running DElab. This problem is not an issue for experienced MATLAB users, because most of the add-on packages for MATLAB are distributed in the same way. With the addition of an advanced help system, integrated into MATLAB's environment, installation process becomes slightly harder. It is our fear that it might turn some novice users away from using DElab. The addition of an automatic installer, as well as uninstaller, should solve the problem. To address the issue of operating system compatibility, we recommend the installer to be done only for Microsoft Windows family of operating systems. People with other OS', such as Linux or UNIX, generally have sufficient knowledge of computers to install DElab by hand. Ultimately, an installer for each operating system can be developed, but the time gone into development of an installer will significantly increase. The digital appendix attached to this report includes installer source files, as well as a test installer written in NSIS. NSIS is a freely distributed tool, and it is also included in the digital appendix.

6.2 Final Thoughts

In order to understand if the new system is more beneficial to users' understanding of the software usage or conceptual knowledge, some testing should take place. As of now, the benefits of the new layout and structure are not quite apparent, since there is not enough content to test the help system with, for example, a study group.

In order to perform these usability studies, interested researcher might consider contacting the same Professors that offered their help to our team. They can provide the guidelines for the correct way such a study should be conducted, as well as provide a group of users that would be willing to participate in the testing.

Even without further testing we are certain that the new help system design addresses the issues that the professors teaching differential equations thought to be most troublesome. This means that most of them were contained in the things that were changed by our team, and when the missing information will be added, the help system will be able to address all of the issues that the students might have with the DElab software or, perhaps, with the various concepts of differential equations themselves.

The research based on the examples of the most successful mathematical packages allowed the team to add information to many areas which were under-researched in the original DElab help system. This ensures that the work that was done corresponds to the industry standards that are in effect now and that the user that is familiar with those products should be able to use the new help system with more confidence and ease.

The purpose of the project was to create a new help system design with a user in mind was successfully accomplished. The final product that the team has produced is a

detailed description of a new help system with a few examples that demonstrate the ideas taken from the research and interviews. With this framework setup it is only a matter of time before the entire help system can be implemented.

Bibliography

Askey Richard.(1997) What do we do about Calculus? First, Do No Harm [Electronic Version]. *The American Mathematical Monthly*, 104(8), pp. 738-743.

Blanchard Paul . (1994) Teaching differential equations with a Dynamical Systems Viewpoint [Electronic Version]. *The College Mathematics Journal*, 25(5), pp. 385-393.

Boyce William E., Ecker Joseph G. (1995) The Computer-Oriented Calculus Course at Rensselaer Polytechnic Institute [Electronic Version]. *The College Mathematics Journal*, 26(1). pp. 45-50.

Brockman, John (1986), *Writing Better Computer User Documentation* Wiley-Interscience Publication

Davis, P.W. (1994). Asking Good Questions about differential equations [Electronic Version]. *The College Mathematics Journal*, 25(5), pp. 394-400.

Etlinger, L. (1974). *The Electronic calculator: A new trend in school mathematics. Educational Technology*, December, 43-45.

Ferrini-Mundy Joan., Graham Karen G. (1991) An Overview of the Calculus Curriculum Reform Effort: Issures for Learning, Teaching, and Curriculum [Electronic Version]. *The American Mathematical Monthly*, 98(7), pp. 627-635

Halmos Paul R. (1994) What is Teaching? [Electronic Version]. *The American Mathematical Monthly*, 101(9), pp. 848-854.

Hubbard John H. (1994) What It Means to Understand a differential equations [Electronic Version]. *The College Mathematics Journal*, 25(5), pp. 372-384.

Hwang, Gwo-Jen (2002) "A conceptual map model for developing intelligent tutoring systems", *Computers & Education* Vol. 40, No. 3 (Apr. 2003) pp. 217-235

Kearsley, Greg (1988), *Online Help Systems* Norwood, New Jersey: Ablex Publishing Corporation

Kember David., Gow Lyn. (1994) Orientations to Teaching and Their Effect on the Quality of Student learning [Electronic Version]. *The Journal of Higher Education*, 65(1), pp. 58-74.

Matthews, D.M. (1996). Mathematics Education: A Response to Andrews [Electronic Version]. *The College Mathematics Journal*, 27(5)., pp. 349-355.

Norman, D, Draper, S Ed. (1986) *User Centered System Design* Hillsdale, New Jersey: Lawrence Erlbaum Associates, Publishers.

Pierce R., Stacey K., (2001) "Reflections on the Changing Pedagogical use of Computer Algebra Systems: Assistance for Doing or Learning Mathematics?", *Journal of Computers in Mathematics and Science Teaching* Vol. 20, No. 2 (Summer 2001) p. 143

Purpose Associates (1998), "Constructivism" available at: http://funderstanding.com/constructivism.cfm (accessed 1 November 2003)

Salisbury, A.B. (1971), "An overview of CAI", *Educational Technology*, Vol. 11, October, pp. 48-50.

Schutte, J.G. (1996), "Virtual teaching in higher education: the new intellectual superhighway or just another traffic jam?" available at: http://www.csun.edu/sociology/virtexp.htm (accessed 13 October 2003).

Seldon, A., Seldon J. (1993). Collegiate Mathematics Education Research: What Would That be Like? [Electronic Version]. *The College Mathematics Journal*, 24(5), pp. 431-445

Slavit D., Cooper K., (2002) "Understandings of Solutions to differential equations through Contexts, Web-Based Simulations, and Student Discussion", *School Science and Mathematics* Vol. 102, No. 8 (Dec. 2002) pp. 380-39

Tapscott, D. (1998), *Growing Up Digital: The Rise of the Net Generation*, McGraw-Hill, New York, NY.

Vom Hofe, R. (2001) "Investigation into students' learning on applications in computer based learning environment", *Teaching Mathematics and Its Applications*, Vol. 20, No. 3, pp. 109-119.

Webnox Corp. (2003), "Computer-Aided Instructions - Definition by HyperDictionary.com", available at: http://www.hyperdictionary.com/dictionary/Computer-Aided+Instruction (accessed 30 October 2003).

Webnox Corp. (2003), "Computer-Based Training - Definition by HyperDictionary.com", available at: http://www.hyperdictionary.com/dictionary/Computer-Based+Training (accessed 30 October 2003)

Appendix A: Interview Questions

List of possible questions to ask differential equations professors:

What are the five or six major concepts in the differential equations course? That is if your students were to come away with an understanding of five or six major concepts what would they be?

Would you recommend a book approach, where the help text would merely mirror what the text book said? If yes, then how would you suggest we should go about picking a book?

How much of background knowledge do you think should be assumed by the help system?

Because there is no universal approach on the amount information that each topic should cover, do you think it makes sense to pursue "less is more" strategy when developing text for help system?

Breaking up the help system outline by differential equations topics or by topics covered in DElab? Would it be a good idea to incorporate both of these access methods (more for HCI professor)?

List of possible questions to ask HCI professors:

What is the current preference in the industry for help system design, is it static or dynamic approach?

Are there some specific criteria for formatting on screen messages? One book we looked in mentioned that fonts with serifs are easier to read, is there a list with proven results of how something should be done?

Appendix B: Professor Heinricher

Subject: Interview with Professor Heinricher
Date: 01/24/04

The questions that were proposed earlier were asked to the professor during the course of the interview, here are the answers to them, and the points that are relevant to answering the six design questions.

1. How familiar are you with DElab software?

Professor Heinricher thought that the DElab software does not work, so he did not know much about its features. The software was then demonstrated to him through the use of screenshots and the review of the package that was installed on his system. A question was brought up by him – Will the software work with the student version of MATLAB? The consequence of this is that he does not advise the student to use it.

2. How much background knowledge do you think should be assumed by the help system, if it is to be used at the undergraduate level?

Professor assumes that the people who will be using this software are the students who are taking the differential equations Course and thus have taken the prerequisite Calculus courses (up to IV). The shift of the student from Calculus courses to the new concepts in differential equations is, in his opinion the most difficult, so we assume no prior knowledge of DE here.

3. Since there are no set standards for the amount information that should supplied to the user by the help system, what approach do you think we should take for choose what goes into help and what does not?

If it is to be used by the College students, it should provide help with homework, at least at Professor Heinrichers opinion. This means that the help system should contain less terminology which is already available from the book, and focus more on what students want to do with the program. For example help on each function should give the students an idea on what they might be looking for (possible solutions, perhaps). Here it is also a good idea to help the students with the interpretation of the result achieved. Sometimes, as Professor Heinricher points out, they come to the right solution, but may not interpret it correctly. It might also contain some examples of doing the computation by hand, explaining how it is done by the software. The system should avoid too many instructions, since it teaches the student to use the program itself, rather than how to solve a particular problem. Some common mistakes should be also included – what to do if something does not look the way it is supposed to. This might be solved, for example by applying the right scale, range, etc. Another feature to help with mistakes might be some pointers that can tell the user if the output that he had received is useless.

4. What organization/outline would you suggest?

Professor Heinricher says that that the best approach towards the organization of the help system would be to follow that of the text book, since it does not confuse the student by throwing the more complicated concepts at them that come out later in the course.

Appendix C: Professor Fehribach

Subject: Interview with Professor Fehribach
Date: 02/02/2004

The following is the results from the interview with professor Fehribach. Professor Fehribach is one of the differential equations professors at WPI. With his experience in dealing with differential equations students and the material Professor Fehribach is a good source of information.

1. How familiar are you with the DElab Software?

Professor Fehribach has used DElab but has not used it in some time. He was able to figure out how to use it and how it works from the information provided with the software. He doubts a typical student would have the same success.

2. How much background knowledge do you think should be assumed by the help system, if it is to be used at the undergraduate level?

Professor Fehribach stated that the help system should assume a background in calculus but no background in computing. This would mean that we would build a help system that is more geared around novice computer users. He also said it would be nice to separate the more novice help and more advanced help. This would allow a more experienced user to be free of basic help associated with beginning users.

3. Information that should be provided by the help system.

Unlike Professor Heinricher, Professor Fehribach did not feel that interpreting results was a problem with students. He felt that getting the program to what the students want it to do was a bigger problem. The student must be able to get the program to work before they can interpret the results. Helping the student get the correct graph or plot is another problem Fehribach brought up. He thought that by having a common errors section for this would make sense.

4. Organization

Professor Fehribach thought Maple's help system was very good. It contains many examples which he feels is good for a help system. He said that examples would help students with plotting. He was not impressed with Microsoft based help systems. He finds that the pop-up windows are annoying and that they should only pop-up when asked for. In his opinion we should stay away from designing a help system similar to the windows style help.

Appendix D: Professor Brown

Subject: Interview with Professor Brown
Date: 01/30/2004

The following memo contains the questions and the answers that our team asked in an interview with Professor Brown from Computer Science department. With Professor Brown's background in Human-Computer Interaction, he was a very value source in spotting and highlighting some of the mistakes present in current help system, as well as suggesting some guidelines that we should follow in designing new help system for DElab.

1. How familiar are you with DElab?

Professor Brown prior to our interview had no knowledge of DElab's existence. Quick demo proved to be very useful, because Professor looked at the whole software in extreme detail, giving comments along the way.

2. What makes help helpful?

This seemed to be the most recurring question throughout the interview. Professor supplied a lot of details. The major points were:

- Text is not a problem in itself
- Font might be a problem, but a very minor one
- Style of writing could be inappropriate, for example novice to expert or expert to novice
- Context sensitivity might be off, for example users asks for A and gets B

Professor also took a very close look at the supplied help, constantly comparing it with the way it is done in MATLAB. A strong point here seemed to be consistency. The user must feel as if they are still in the MATLAB, therefore all visual elements must stay the same way as they are in MATLAB(for both, help and DElab's interface). From this point on, the relevance of other questions was diminished because they were answered by one answer "consistency."

Looking at DElab's help Professor Brown made a few interesting points, which, we feel, should be addressed in the final design of the help system.

- DElab topics should be coded so that they are shown on the right, in the topic access tree, with the current one being highlighted.
- Make names clickable (associative links)
- Structure into smaller chunks and go into chunks directly, with chunks laid out in the topic access tree in the same way that MATLAB does it
- Use of HTML for help pages themselves would be appropriate, because MATLAB makes use of HTML based pages

- Font can be changed to one that includes serifs, but that would mean trading off consistency for speed where speed is not that important
- Whenever MATLAB provides links to different topics, it gives a short summary for each link, so that the user will know what to expect on that page
- All pages must have a standard applied to them, to make them consistent with each other, as well as the whole help system
- Any references to window's titles must be identified clearly, because the user needs to locate those windows
- Associate a color with DElab help to make it stand out from MATLAB. Currently MATLAB's top bar which displays current topic and shows link to next topic and previous topic has the same color all throughout the help. Having a different color for all DElab pages would be a good idea, because user will immediately know if they are looking at DElab's function of MATLAB's function
- Make links work!
- No need for filenames – use standard English just the way MATLAB does
- Check all the text to make sure it is correct English
- Whenever there is more information in the book available provide page number, or section number of the chapter

Appendix E: Value Matrix

Subject: Software Help System Evaluation
Date: 1/27/2004

Visual Components:

The "Visual Components" section of Value Analysis consisted of four different subsections: fonts, toolbar, content pane, and navigation pane, help window's placement on the screen and pop-up windows and presence of secondary windows. All of the evaluations were based on a five point score rubric, summarized in the table below.

NA	Poor	Below average	Average	Better than average	Excellent
0	1	2	3	4	5

Each score section is self-explanatory. For less obvious criteria such as fonts we tried to see how much of them conform to the standard set forth by the help system design research done by Microsoft, for example the absence of a font would represent 0. Very small unreadable fonts would correspond to 1, readable but incorrect font selection would mean 2, and so on all the way to the well readable size and correct font selection (fonts with serifs are desired).

Maple 8

Fonts: 2 – the fonts were readable, however the font selected was lacking serifs, which made it harder to read.
Toolbar, content pane, navigation pane: 3 – All of the three panes mentioned were present, but they were somewhat unintuitive due to poor button icon selection. For example an arrow with a line does not necessarily mean "Go to index." Also panes were not well organized on the screen; the horizontal arrangement leaves too much empty space and takes away from the main help area.
Help window is not in the way: 2 – it can be moved to not cover the main screen where you type in the commands, but its initial position fully covers the main window.
Pop-up windows and secondary windows: 1 – there were some pop-up windows, for example the tutorial window, but those windows would usually keep generating more new windows which was highly annoying.

MATLAB 6

Fonts: 2 – for the same reason as Maple 8, absence of serifs in the font.
Toolbar, content pane, navigation pane: 5 – excellent layout of all of the panes, the side bar has the content and the major chunk of the window is occupied by the help text

itself. Also the navigational buttons are consistent with Internet browser looks, which makes them easier to understand.
Help window is not in the way: 4 – Help window initially is out of the way, but not entirely. None of the programs that we looked at implemented a system similar to Microsoft Office help, where the help window opens to the side of the application window. This makes it easy to read the help and use application at the same time.
Pop-up windows and secondary windows: 0 – there were no pop-ups at all, and no secondary windows. Everything is located within one window.

MathCAD 2001

Fonts: 5 – very well selected font with serifs.
Toolbar, content pane, navigation pane: 5 – the layout of these three bars was very similar to the layout in MATLAB; however MathCAD utilizes Window's standard help interface which is also very easy to use.
Help window is not in the way: 4 – help window was out of the way, but it was not offsetting the main application to ensure no overlaps.
Pop-up windows and secondary windows: 1 – for the most part there were no pop-ups or secondary windows.

Maple 9

Fonts: 2 – same font selection as in Maple 8
Toolbar, content pane, navigation pane: 4 – a much better layout in comparison to version 8, however the table of contents looked awkward without the plus (+) signs to signify expanded or collapsed topics.
Help window is not in the way: 4 – the new help window does not cover the application window. There are some overlap areas but the window can be easily moved out of the way.
Pop-up windows and secondary windows: 1 – almost no pop-ups or secondary windows, only tutorial used new window approach.

Mathematica 5

Fonts: 2 – same as font selection in Maple 8 or MATLAB.
Toolbar, content pane, navigation pane: 3 – The layout was very similar to Maple's version 8 layout with some minor additions which made it more useful, for example the presence of the index tab.
Help window is not in the way: 4 – same as Maple 9.
Pop-up windows and secondary windows: 0 – no new windows, and no pop-ups.

Features:

All of the rows in this part of the value matrix are rated equally important, since they all help the user to navigate the system faster and more efficiently, each in its own

way. All of the numeric values that were assigned correspond to the standards for the particular parts that were described in the Microsoft research, as well as our own unbiased impression of their functionality.

Maple 8

Presence of Index: 0 – The Index is not present.
Presence of Table of Contents: 4 – Table of Contents is well organized, however it is not configured in the tree structure, which has already became a standard for this element of the system.
Presence of Search Function: 4 – Overall good Search Function, however it does not provide all of the search possibilities that MATLAB and MathCAD provide.
Associative Links: 5 – A lot of useful Associative Links that point to the help files that are relevant to the subject.
Quick Reference: 0 – Quick Reference is not present.

MATLAB 6

Presence of Index: 5 – Follows all of the standards that were mentioned in the Microsoft research paper.
Presence of Table of Contents: 5 – Well organized, uses the currently standard organization, even has icons for different types of content that is presented.
Presence of Search Function: 5 – Excellent Search. We were able to find anything that was needed.
Associative Links: 5 – Same as in Maple - useful Associative Links are present that point to the help files that are relevant to the subject.
Quick Reference: 3 – Quick Reference is present for most terms that need explanation.

MathCAD 2001

Presence of Index: 5 – Same as MATLAB.
Presence of Table of Contents: 5 – Same as MATLAB, minus the icons.
Presence of Search Function: 5 – Excellent Search. We were able to find anything that was needed.
Associative Links: 5 – Same as in Maple and MATLAB - useful Associative Links are present that point to the help files that are relevant to the subject.
Quick Reference: 4 – Quick Reference is present for all terms, up to the lowest possible degree.

Maple 9

Presence of Index: 5 – Same as MATLAB and MathCad, gets points off for not listing the help topics by typing in just the first letter.
Presence of Table of Contents: 5 – Improvement from Maple 8 – now uses the standard tree structure, which caused us to reduce points for this element in the first place.

Presence of Search Function: 4 – Same as Maple 8.
Associative Links: 3 – A huge step down from the Maple 8 system, probably came with the introduction of the Index.
Quick Reference: 0 – Quick Reference is not present.

Mathematica 5

Presence of Index: 3 – Is present, but implemented awkwardly, reminds of the structure of Table of Contents for Maple software.
Presence of Table of Contents: 4 – Same as Maple 8.
Presence of Search Function: 4 – Same as Maple 8 and 9.
Associative Links: 2 – The Links are almost nonexistent, the creators of the software assume that the user will find the information with the other means provided.
Quick Reference: 0 – Quick Reference is not present.

Functionality:

Conceptual information we deemed to be the most important of the functionality areas. The conceptual information is the part of the help system that is going to help the person learn and fully understand the material. Helping someone get the solution is not good enough. It is also important that they understand the solution. This is what we looked for when evaluation the help systems.

Interactive help functions we ranked as next important in the functionality portion. We looked for windows in the help that allowed the user to change variables and to see how it changed graphs and or results. We also looked for help windows that allowed the user to execute functions in the help window.

Context sensitive help is not quite as important because it is basically just the way you access the help. It is important to have several straightforward ways to access the help system to prevent user irritation.

Maple 8

Conceptual Information: 2- Most of the help just explained what each function does and what each variable stood for. The help contained graphs and charts but lacked detailed explanations of what each represented. It also failed to touch on what the significance of each function was and what the output meant.
Context Sensitive Help: 4- provided good context sensitive help. Maple8 allows you to right click on the function for help or to access help through a keystroke. Also contains the standard help button in the program itself.
Interactive Help Functions: 3 – Maple8 does a good job of incorporating interactive help functions. The help in Maple allows you to take functions directly from the help and paste them into the Maple window. It also allows you to run worksheets in the help window. It does lack truly interactive examples where you can plug in different variable values and get different results.

MATLAB 6

Conceptual Information: 4- MATLAB does a good job with the conceptual explanations of functions. Through the use of associative links and quick reference functions it provides a solid background for any mathematical process.
Context Sensitive Help: 5- The help in MATLAB has all the features that Maple8's context sensitive help. MATLAB also allows the user to have a sort of quick help in the actual application window.
Interactive Help Functions: 2- MATLAB lacks any truly interactive help functions and does not allow you to run functions in the help window. It does allow for the copy and pasting of functions from the help to the application window.

MathCAD 2001

Conceptual Information: 3- Descriptions of help functions are not as detailed as MATLAB but still does a good job. Overall it lacks the explanations of solutions and information on interpreting graphs.
Context Sensitive Help: 5- Help can be accessed in all the same was as MATLAB.
Interactive Help Functions: 5- Has a truly interactive help. Allows you to change variables and observe the changes in graphs and results. It also allows you to run functions from the help window.

Maple 9

Conceptual Information: 2- Not much different than the help system in Maple8.
Context Sensitive Help: 4- Comparative to Maple8.
Interactive Help Functions: 3- Same as Maple8.

Mathematica 5

Conceptual Information: 2- Lacks detailed descriptions of functions and how to use them. Mathematica does not have important information on explaining results or interpreting graphs.
Context Sensitive Help: 4- Similar to Maple8 and Maple9, it lacks the in application help feature.
Interactive Help Functions: 5- This help system also contains truly interactive help functions. Has help examples that are very similar to MathCAD's interactive functions.

Value Analysis											
Criteria	**Weight**	Maple 8		MATLAB 6		MathCAD 2001		Maple 9		Mathematica 5	
		Value	**Total**	Value	**Total**	Value	**Total**	Value	**Total**	Value	**Total**
Visual Components	20		**41**		**59**		**71**		**58**		**45**
Fonts	10	2	4	2	4	5	10	2	4	2	4
Toolbar, content pane, navigation pane	35	3	21	5	35	5	35	4	28	3	21
Help window is not in the way	25	2	10	4	20	4	20	4	20	4	20
Pop-up windows and secondary windows	30	1	6	0	0	1	6	1	6	0	0
Features	40		**104**		**184**		**192**		**128**		**104**
Presence of index	20	0	0	5	40	5	40	4	32	3	24
Presence of Table of Contents	20	4	32	5	40	5	40	5	40	4	32
Presence of search function	20	4	32	5	40	5	40	4	32	4	32
Associative links	20	5	40	5	40	5	40	3	24	2	16
Quick reference	20	0	0	3	24	4	32	0	0	0	0
Functionality	40		**108**		**144**		**160**		**108**		**132**
Conceptual information (examples, graphs)	50	2	40	4	80	3	60	2	40	2	40
Context sensitive help	20	4	32	5	40	5	40	4	32	4	32
Help functions interactively	30	3	36	2	24	5	60	3	36	5	60
Totals			**253**		**387**		**423**		**294**		**281**