Mental Models for Introductory CSC Concepts

Jennifer J. Wheeler and Erika Rogers

Abstract - It is hypothesized that the development of mental models for a concept can lead to improved learning. The purpose of this paper is to identify, quantify, and analyze the mental models of students beginning to program in the UNIX environment. By focusing on a basic concept, this study abstracts current mental models from students and hypothesizes ways of improving these models. Ultimately, the results of the study will be incorporated into the design and implementation of a multimedia tutor. This tutor will attempt to teach people how to develop mental models in order to improve their learning.

Introduction

The complicated processes of today’s technology often have steep learning curves. To improve these curves, new techniques need to be developed to accommodate these complexities by improving understanding and performance.

When a novice user sits down at a computer, he/she can feel intimidated or confused. Its technology appears so much more complex than traditional educational, research, and office tools. For these users, the thought of this machine actually making their tasks easier seems impossible. In this era of computing, this kind of fear prevents people from acquiring the skills needed to function competitively and productively. Since computers are all around us, a very large part of our population is familiar with and uses computers in their everyday lives. Are they comfortable with them? Do they really know how a computer carries out its tasks?

Even computer scientists can feel intimidated by some computing concepts. New computer science freshmen, even if experienced with personal computers, can feel uneasy with the tasks ahead of them in the work towards their degree. For those students programming in the UNIX environment, that feeling often is increased when faced with a non-graphical interface for the first time. UNIX is a command line operating system and a user cannot rely on the desktop metaphors with which many of us are familiar and comfortable.

One approach to making new users more comfortable with UNIX is to learn how to build a better internal understanding and representation of UNIX. Cognitive science calls these internal understandings mental models. Donald Norman has described them as “the model people have of themselves, others, the environment, and the things with which they interact. People form mental models through experience, training, and instruction.” (Preece, pg. 130) This paper explores the idea that the learning curve for UNIX can be improved if instead of forcing UNIX with ‘rote memorization,’ mental models can be used to help students formulate an understanding of not only how to work with UNIX, but how UNIX works. When a complete understanding of a structure and process is intact, further exploration and skill acquisition can occur more rapidly and accurately.

It is expected that by developing a process by which mental models for computing concepts can be extracted and improved upon, other computing concepts can be approached in a similar manner. “Much has been written about why designers need to develop interfaces that will shape a user’s mental model, but there are very few suggestions as to how to achieve this. Part of the problem is that we simply do not know enough about how people construct and use models of computer systems.” (Preece, pg. 137) This study develops an experimental method for accomplishing this goal.

We suggest that by understanding current models of novice users, these models can then be improved upon and realized in a multimedia tutor. “With interactive multimedia programs the learning process becomes active, not passive, and it ensures that users are doing, not simply watching.” (Stemler, pg. 340) By building appropriate mental models with the use of multimedia, the chances for an improved learning curve and understanding should be greatly improved. “By analyzing the process through which users acquire beliefs, correct or otherwise, about a system, it is hoped that interfaces can be designed that support the acquisition of appropriate user mental models.” (Preece, pg. 124)

Background

A mental model is a person’s internal representation of a process or an idea. Often these models are not completely developed or are misleading. This can lead to poor performance and frustration.

“Mental models are either analogical representations or a combination of analogical [picture-like images] and propositional [abstract, language-like statements] representations.” (Preece, pg. 131) Using this theory, our study attempts to extract user’s mental models by studying both verbal and visual clues. This approach takes into
account different types of learners and thinkers. The current models must be approached before new or improved ones are developed because "the general assumption is that people do use some type of model, but that is often incomplete and vague." (Preece, pg. 137)

Ronald Baecker has stated, "It is generally assumed that users apply mental models in trying to understand and predict system behavior, and that their success in using a system depends on how well their mental model corresponds to a model represented in the system design." (Baecker, pg. 581) In many areas of computer use, these mental models are often incomplete or incorrect for users. Therefore their experience causes them frustration and may inhibit them from exploring and utilizing a computer's potential. The reason these mental models of computing are so difficult to develop is the very nature of the computer. It is a complex, electronic system that takes years of training and experience to truly understand. An everyday user does not need to understand the workings of the system but they do need to have a concept of how it is organized and how it functions. Users probably find it extremely difficult to abstract a mental model of a computing task because they cannot make these kinds of comparisons to the real world. To abstract mental models necessary for successful use, these comparisons must be made.

It is postulated that one of the reasons users have difficulties navigating and using directory structures (especially in UNIX) is because the mental model is based on metaphors that are too simple for the complex task. The metaphor used on the desktop today is that of a file drawer. Files go into folders (the directories) and are stored. The computer environment is much more complex and so this metaphor cannot cover the entire scope of the mental model needed. Also, this system does not give a good understanding of applications. Used in the UNIX environment, the metaphor has hardly any effect on the user's mental modeling since files and directories can appear exactly the same. (Windows and Macintosh environment use icons to distinguish between the two.)

Experimental Design

The purpose of this study was to both determine if a need for improving UNIX skill acquisition is needed for beginning computer science students and to develop a method for extracting mental models in a way to accurately develop ways of improving them.

The key element in the design of the methodology for this study was to focus all questions and procedures around the idea of extracting the current mental models from the subjects and then evaluate how these models were affecting skill level and performance. The secondary focus of the study was to acquire evidence that improvement of UNIX skill acquisition was necessary.

Subjects and Materials

Novice computer users have to learn to work with the file structure system of computers, though it often seems that they learn to work around it instead. Many never fully understand the differences between directories, files, and applications.

Subjects for the study included students in beginning computer science classes in which fundamental programming concepts are taught. In these courses, students learn to program in the UNIX environment, but are never actually taught UNIX in the curriculum. They are forced to struggle through the concepts on their own which often leaves them with poorly developed mental models and unsatisfactory performance. Students participating in the study were at the end of their first quarter and so presumably had at least three months' experience with UNIX.

One hundred thirty-five students participated in the quantitative, survey portion of the process. The four page survey attempted to assess skill and comfort with UNIX while also attempting to extract the mental models of directory structures.

From these students, six volunteered to participate in the hour-long qualitative portion of the process. These students performed tasks with a directory structure on a Macintosh and a UNIX telnet session. A simple sample directory structure was set up on both systems in order to perform navigation tasks. The structure consisted of seven directories, three files, and one application in a three level hierarchy. Some of the tasks they performed were to copy files, add and delete directories, edit files, and simple navigation tasks.

Phase I Procedure

In the first phase of the study, we wanted to get a large-scale view of the subjects' understanding of the UNIX environment. A four-page survey was developed for students to complete in ten minutes in a classroom setting. The survey was meant to extract information on mental models, UNIX proficiency, confusing aspects of directory structures, and self-assessment regarding UNIX usage.

Part one of the survey was comprised of questions to look at the background and self-assessment of subjects. One question asked how long the student had worked with UNIX. Two asked for an identification of easy and confusing tasks. Five questions asked for a self-assessment through a rating system regarding competence, confidence, interest, and UNIX navigation.

Part two of the survey focused on extracting mental models and understanding of directory structures. People develop very different kinds of mental models. Some are visual, some are structural, and others are process models. In order to try to extract all of the different types of models, subjects were asked to define in words and/or pictures, the relationship between directory structures and their
components. Subjects were asked to “Please describe or define the relationship of files, directories, and program applications in words and/or pictures using examples” and then to draw or explain “/foo/widget/smith.txt.” Research into how directory structures are presented to students suggested that the tree-structure would be the representation most often used by students in these descriptions.

The final section of the survey performed a more restrictive extraction of mental models as well as assessing basic skill level. Subjects were asked to answer seven quiz-style UNIX questions (Figure 1). Five of the seven asked the students to answer by showing on a diagram where a command would take them. Three different diagrams were presented to find out which one was used most often. Two of them were tree structures while the third represented the ’desktop’ metaphor used in the Windows and Macintosh environments. Figure 2 shows the diagrams used.

1. Place a 1 on one directory
2. Place a 2 on one file.
3. If my path is /home/proj/help, put a 3 on the directory where I am in after I execute the command cd ../..
4. If I am in assign1.c, put a 4 on the directory represented by /
5. If I want a subdirectory named schedule in work and I am in work, what command do I execute?
6. If I am in Home, and execute the command cd fun, put a 6 on where I am.
7. If I am in work, what commands do I execute to remove help (if help is empty)?

Figure 1: Survey Quiz Questions

<table>
<thead>
<tr>
<th>Home</th>
<th>fun</th>
<th>proj</th>
<th>work</th>
</tr>
</thead>
<tbody>
<tr>
<td>assign1.c</td>
<td>help</td>
<td>lab1.txt</td>
<td></td>
</tr>
</tbody>
</table>

diagram 1

<table>
<thead>
<tr>
<th>Home/</th>
<th>fun/</th>
<th>work/</th>
</tr>
</thead>
<tbody>
<tr>
<td>proj/</td>
<td>assign1.c</td>
<td>lab1.txt</td>
</tr>
<tr>
<td>help/</td>
<td>help</td>
<td></td>
</tr>
</tbody>
</table>

diagram 2

<table>
<thead>
<tr>
<th>Home/</th>
<th>fun</th>
<th>work</th>
</tr>
</thead>
<tbody>
<tr>
<td>proj</td>
<td>assign1.c</td>
<td>lab1.txt</td>
</tr>
<tr>
<td>help</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

diagram 3

Figure 2: Survey Diagrams

Phase 1 Results

The results of the survey show that there is a need for improved UNIX skills among beginning computer science students. The survey also provided information on ‘problem areas’ as well as exposing some students’ current mental models.

The majority (75.5%) of students surveyed had less than three months’ experience with UNIX. The most commonly cited aspects of UNIX that seemed easy to the subjects were email, the similarity of commands and file structures to those of DOS, and navigations. Editors, the command line interface (as well as remembering commands), and navigation were cited as difficult aspects of UNIX. The results of these responses in part one were not unexpected although the fact that navigation came out as difficult and easy leads us to believe that the subjects are grasping concepts at different rates and that subjects are, in general, having difficulties with UNIX concepts. The consistent references to DOS suggest that the GUI interfaces of Windows environments do not improve an understanding of directory structures. On the average, students did not feel very confident with UNIX and students rated themselves neutral on the self-assessment part of the survey.

Part two of the survey resulted mostly in text descriptions and/or tree diagram representations. The tree diagram is generally used to teach directory structures. We assume that they drew the tree structures because that is what they are taught. It does not necessarily reflect how they are really thinking about the problem. However, a second model was identified, which we call the “container model.” In contrast to the traditional hierarchical tree model, subjects using the container model seemed to think of part of the process as going into and out of structures in order to move around, store items, and retrieve items. By analyzing subjects’ words and diagrams such as those shown in Figure 3, the container model emerged as another expression of the mental model of subjects.
Mental models were further explored in the final section by observing which diagrams were used to respond to the 'quiz' questions. The majority of subjects (61%) responded on diagram 1—a tree structure. This result is somewhat difficult to evaluate since this may have been chosen most often because it was shown first.

The final portion of the survey (Figure 1) was aimed at skill assessment. It was surprising how poorly many questions were answered since the concepts were very simple and this is the environment the subjects had been programming in for at least two months. Figure 4 shows the results of questions one through seven. The most basic questions (1, 2, and 6) did get a high percentage of correct responses but not high enough considering their simplicity. Questions 3 and 4 dealt with the . and / concepts. The subjects did not seem to have a grasp of these navigation tools and many did not even know what they were. Question 5 should have received a higher percentage than it did since the syntax for the "mkdir" command was all that was needed for the question. This seemed more like a memorization issue than anything else did. The final question was the most difficult since it involved multiple commands and more complex navigation. The 21% result is the most obvious proof that students need more help with UNIX and navigation concepts.

Phase 2 Procedure

After the surveys were compiled and analyzed, phase two was developed accordingly. The points we wanted to explore further were the problem areas, the container model, and usage of commands. In phase two, each subject worked one-on-one with the investigator. Whereas the survey was looking for an overall skill level and understanding, the interviews provided a direct look at individual understandings, problems, and representations. After obtaining human subjects approval, six interview sessions were conducted, each of which took approximately 40 minutes and were tape recorded and then transcribed.

Each subject was first asked how long he had been working with UNIX. Next, the subject was given a 'think aloud' lesson to improve the quality of the tape recordings.

After this initial part of the interview, the subject sat at a UNIX workstation to perform an exploration of the sample directory structure as well as a number of navigation tasks. He was given a 'cheat' sheet of basic UNIX commands so that the focus would not be on memorization of commands but rather on the understanding of directory structures.

Ten minutes were allotted for the subject to explore the directory structure while both 'thinking aloud' and drawing a picture of the directory structure. Having the subject perform the task in this way provided various observations. Extracting the mental model of the subject in this part of the experiment combined an observation of visual and auditory data. This way the mental model could be more accurately interpreted. By having the subject produce a visual representation while also recording key words and phrases as the directory structure was explored, two sensory inputs provided insight into how the subject viewed and understood the directory structure. The sessions were logged in order to provide a record to later compare how different subjects explored the structure, which commands and concepts were avoided, and where problems occurred.

Next, the subject was asked to perform tasks on the directory structure that included file structure manipulation, use of applications, and also forced the subject to use particular commands. This allowed for further assessment of skill level, and conceptual understanding.

Then the subject did an exploration of the same directory structure (with a different naming scheme) on the Macintosh. The same procedure of 'think aloud' and pictorial representation was used to compare performance and understanding.

Finally, the subject was asked to put each of the ten 'cheat' sheet commands into categories (Figure 5). The
"Build and Modify Space", "Move in Space", and "View Space" categories were selected as representative of how typical UNIX users might think about the command. "Touch Space" and "Extract Space" were identified as possible auxiliary categories but were not expected to be commonly used. The purpose of having the subjects categorize the commands was to open up one more venue from which to draw conclusions about the mental models. It was thought that the category in which the subject placed the command could give more insight into his mental model. The auxiliary categories were decided upon because they were logical ways to interpret the commands and could give some insight into a different way of thinking about them. Each subject was given five categories but instructed to create a new one if he thought that a different category was more appropriate.

<table>
<thead>
<tr>
<th>Command Category</th>
<th>cat</th>
<th>cd</th>
<th>cp</th>
<th>ls</th>
<th>mkdir</th>
<th>more</th>
<th>pwd</th>
<th>rm</th>
<th>rmdir</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build and Modify Space</td>
<td>1</td>
<td></td>
<td>5</td>
<td></td>
<td>6</td>
<td>4</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move in Space</td>
<td></td>
<td>5</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Touch Space</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extract Space</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>View Space</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
* Copy and move in space  |     |    |    |    |       | 1    |     |    |       |
* View location of space  |     |    |    |    |       |      | 1   |    |       |
* Delete space            |     |    |    |    |       |      |     | 1  | 1    |

* Categories defined by subject 6

Figure 5: The command categories

Phase 2 Results

The concept of the container model was reinforced with phase two of the study. Subjects used words like ‘in’ (suggesting a container model) more often than ‘under’ (suggesting a tree or hierarchical model). This suggested that although the subjects learned with a tree diagram (as seen in phase 1), they either weren’t using it or did not know how to make it useful. One subject who was very experienced with UNIX was particularly interesting to observe. He used both container words (e.g., “in”) and hierarchical words (e.g., “under”). This suggested that the more experienced user was actually using two different models at different times. While describing the directory structure he seemed to be using the tree model, but while actually navigating through, he referred to container concepts.

Five subjects drew a tree representation while the sixth wrote out the UNIX description. They were all able to correctly represent the directory structure, but even with the correct tree diagram, they had trouble navigating through the structure.

All the subjects performed well on the navigational task assigned to them. However, five out of the six could not perform a task requiring the command ‘cd ../../maria’. Also, subjects often performed tasks in an inefficient manner using two or three commands when one was required. Often this was because subjects did not make use of the flag options available with the commands.

The most glaring problem was moving from subdirectories to parent directories. Most used full path names of the directory rather than utilizing the . and .. directories. While this worked for the three levels of the test hierarchy, the use of the full path name or always returning to home appeared to be very time consuming.

In terms of the categories (Figure 5), only one subject categorized the commands in the manner we expected. The results of the categorization suggest that the full functionality of some of the commands is unknown or unused. For instance, ‘cat’ was most often placed into the “View space” category when its function can be to combine files and create files. The most skillful subject, who utilized both tree and container models, placed “cat” and “more” into the “Touch space” category. Although “Touch space” was not expected to be used, this subject’s interpretation suggests one way to describe the quick view of the file.

Discussion

The final analysis of this study has resulted in the idea that two types of mental models should be used together in order to increase productivity and efficiency with the UNIX environment. This is because when working with the UNIX directory structure, there are two kinds of thinking taking place. In order to have a proper understanding of the structure, a topological model represented by a tree/hierarchical structure should be presented. But, in order to understand the movement and functionality of the structure, a process model represented by a container model should be used.

This study has clearly shown a need for more in-depth UNIX instruction while also exposing problems with command usage, navigation, and accurate development of mental models. While this study focused on the very basic commands and concepts with UNIX, the procedure and results can be used to expand into more aspects of UNIX and other computing topics.

Often, both sets of subjects had inefficient and unproductive usage of the UNIX commands. One of the problems is that the students are never taught the "best" way to perform a task. Since they are given a book to read through, they are shown many ways to do things but not the way they should be doing them. Part of the problem is that students seem to have an aversion to using the flag options with the commands. They either do not know about all the options or are afraid to use them. In the survey, many students stated that they were afraid of destroying items that were not meant to be destroyed. With some commands,
options exist which prompt the user before finally deleting files. Subjects also stated that they had difficulties distinguishing between files and directories. With flag options, the elements of the structure can be distinguished. Finally students used inefficient command sequences. This was especially clear when subjects were presented the task of removing a directory. Because they were unaware of recursive deletion, they would go into the directory and remove all files and then delete the directory. (This problem was first exposed in the survey. Some students did not know a directory had to be deleted with "rmdir" rather than rm.)

The study also gave some insight into how students think UNIX works -- their mental model of UNIX. Most often, when learning UNIX, they are exposed to the tree hierarchy as well as their previous experience with Windows and Macintosh environments. We believe this combination of conflicting models causes confusion and frustration. The visual representation vacillates between the desktop metaphor and the tree structure. When one is used incorrectly, the student's performance and accuracy is effected.

By teaching UNIX in a way that develops the appropriate model for each task, the confusion and high learning curve students encounter could be improved. While the topological model should remain as a tree structure, the process model should be similar to the desktop metaphor but focus on the containment qualities of the metaphor rather than the physical representation of folders and files. We believe that building mental models of the UNIX directory structure with these representations would best prepare students for understanding and using UNIX. By using a tree hierarchy as the visual representation, the student has a simple, two-dimensional picture of what the directory structure looks like. This is essentially what is used currently. But by concurrently developing the process model using the container model, students develop a better understanding of storage and navigation. The container model would facilitate the understanding of moving out of a directory to the parent directory before moving to a directory on the same level. This particular navigational aspect is one of the reasons the desktop metaphor causes confusion. In the desktop metaphor, two directories can be opened simultaneously so there is no need to move in and out of directories. Also, with the desktop metaphor, the dragging feature has no equivalent in the UNIX environment in terms of the mental model. The development of this dual representation would allow students to accurately understand different features and tasks in UNIX. Although this instructional approach would take more effort than is typical of basic tutorials or books, it would be worthwhile if it helps student focus more on programming rather than struggling with UNIX.

In order to provide support for this additional effort, we believe that multimedia can be used very effectively to produce a tutor that emphasizes both mental models concurrently. Some key elements of multimedia are graphics, animation, video, and sound. With vocal and moving descriptions, the process model can be developed more successfully because seeing a process provides a better understanding than reading about one. The second key feature of multimedia for this development is interactivity. This allows the student to participate in "hands-on" learning. The interactivity would also provide a way of encouraging the appropriate model for a particular task.

Summary and Conclusions

Both phases of the study have shown a definite need for more UNIX instruction than the "learn as you go" technique used now. Instruction is needed so that the students are both aware of the flexibility and power of UNIX as well as how to use it most efficiently and productively. UNIX is the environment students work in and therefore needs to be made easier to learn and to use.

The next goal for this study is to use the results to develop a standalone interactive multimedia tutor that will have the following elements:

- While explaining commands and processes, have a tree structure in view at all times but explain the commands using a real world metaphor that encourages the containment model to be developed.
- With each command, make the student aware of the options and the best way to use them.
- Give students a "security blanket" by showing them the protective features in UNIX.
- From the beginning, show and use the ., .., and * concepts for more efficient navigation.
- Develop a process model that describes navigation and structure manipulation using the container concept. Simultaneously develop a topological model of the structure using a tree/hierarchical structure and provide a mechanism for integrating the two views.

References

