

What's Involved with Moving from One 3-D Constraint-based CAD System to Another?

Abstract

During the early 1990's, many educators and professionals in the field of engineering and technical graphics were faced with the movement from 2-D Computer-aided Design (CAD) systems to 3-D modeling systems. More recently, many of these individuals are faced with a new transfer of training issue, this time surrounding the move from one 3-D CAD system to another. The purpose of this research was to test the hypothesis that, with minimal training, users can transfer their established high level modeling strategies between packages (thus preserving an important element of efficiency and effectiveness). This transfer will take place once they have mastered the basic syntax necessary to execute their strategies.

The results of the study clearly showed the initial conflict when there were syntactic differences between the software tools in executing a specific modeling strategy. A majority of these conflicts, however, did not re-occur once they were resolved at the time of first encounter. There was also clear evidence as to the ease in which original higher-level task strategies could be preserved on the new system. This study demonstrated clear differences between the transfer of syntactic and semantic level knowledge from one 3-D modeling system to another. The results of this study also have implications for the design of training and educational programs, pointing to the importance of delineating between software package-dependent and independent skills and knowledge.

Introduction

In the 1980's and 1990's, many educators and professionals in the field of engineering and technical graphics were faced with the movement from drafting-based production of engineering drawings to 2-D Computer-aided Design (CAD) systems (Arnold & Bessant, 1983). In the 1990's there has also been considerable movement from 2-D CAD systems to 3-D modeling systems. A common issue centered around the organizational issues of moving from 2-D to 3-D CAD systems, including training issues (Klein, Hall & Laliberte, 1990). Currently, many of these companies are faced with a new training issue, this time surrounding the move from one 3-D CAD system to another. This issue has become particularly pronounced with the increased mobility of professionals and the emergence of more cost-effective (thus, more widespread) Windows-based modeling systems.

Studying the training issues of moving from one constraint-based 3-D modeling system to another will not only provide pragmatic help to industry managers, but also to educators trying to better devise instructional strategies which better prepare students with a robust and flexible structural knowledge of constraint-based modeling systems. There is an excellent chance that if students are using 3-D modeling software five years after graduation, it will be a different modeler and/or a different version than what they used in school. The goal is to better prepare students to be life-long learners.

Transfer of training

The standard use of the term transfer of training refers to the transfer of skills learned in a training class to the workplace (Baldwin & Ford, 1988). Here, in this study, the term is used in a slightly different way. Instead of transfer of 3-D modeling skills learned in the classroom to the workplace, it is the transfer of skills from one modeler in the classroom to another modeler used in the classroom. The hope is to be able to infer from this what a student might face if the second modeler was one being used on the job after graduation. In this experimental setting, we are ignoring other key factors involved in transfer (i.e., trainee characteristics, such as creativity and intelligence; and work-place characteristics, such as peer and supervisor support) and focusing exclusively on the success of transfer of learned materials from one 3-D modeling system to another.

Transfer of training is often thought of exclusively in terms of motor or psychomotor tasks, such as driving a car or playing baseball. In an information-based workplace, the cognitive aspects of training take on increasing importance. Learning the operation of computer software almost exclusively involves the development of cognitive skills. The differences between cognitive and motor skills means that the rules governing the transfer of motor or psychomotor skills cannot be uniformly applied to the transfer of cognitive skills. Evaluating to effectiveness of computer software training involves, in part, understanding the development of a mental model, or schema, surrounding the software (Carroll & Olson, 1990). This mental model provides an underlying structure for both planning the next action and understanding the response of the system to a user action.

In understanding the transfer of skills from one computer program to another, classic transfer of training principles do hold for more habitual or routine tasks with the software. For example, if there is a routine command in Software A for which the same keypress or menu pick achieves the same result in Software B, then there will be a strong positive transfer of training from software A to B. On the other hand, if the very same keypress or menu pick prompts a completely different action in Software B, then there will be a negative transfer of training. That is, thorough practice with Software A will actually impede mastery of B.

Shneiderman's OAI model

Understanding the impact of knowing Software A on learning Software B for tasks that are not habitual or routine becomes more difficult. One approach is to step back and first look at the larger task that the user is trying to achieve with the software tool. From this perspective, the impact of the mental model formed of Software A on the learning of Software B may be better understood.

Shneiderman (1998) has proposed a model of human-computer interaction, the Object-Action Interface (OAI) model, which provides a structure for understanding how users transfer real-world tasks to the computer interface of a software package. Part of this transfer process involves the translation of higher-level task strategies (the semantic level) into software specific commands (the syntactic level). Previous research has indicated that many of the most popular 3-D modeling packages all contain commands/tools which support the same higher level modeling strategies for modeling simple parts (Wiebe, 1999). In contrast, all

of the packages have different interface elements, which create different syntaxes for achieving these higher level goals.

This research is an initial look at methods which might be used to better understand the process of learning a new software package to perform a task previously performed on a package the user had some degree of mastery over. More specifically, how successfully can the user take higher-level (semantic) task strategies to a new, functionally similar but syntactically dissimilar 3-D modeling system?

Study Design

An upper level engineering graphics course in 3-D constraint-based modeling was used as the sample in this study. The constraint-based modeler used in this course was Pro/ENGINEER (Pro/E) v20. After 12 weeks of a 15 week course, students were asked to model the valve body shown in Figure 1. At this point in the semester, students had moved beyond simply memorizing commands on the software. They had committed to memory the basic, declarative knowledge concerning the location and function of basic commands and had brought these commands together into procedures to be used to solve modeling problems (Anderson, 1982). Though most of the techniques needed to model this part were learned in the first 5 weeks of the semester, a new technique, radial patterning, was used to create the three tabs spaced 120 degrees apart on the body. After demonstration of the radial patterning technique on an example model, students were instructed to create the valve body however they saw fit. Files recording all of the Pro/E commands used by the students (called 'trail files') were collected up from the students when they finished their model. Time for model completion was also recorded.

During the next class period, it was announced that students would have an opportunity to try out a new constraint-based modeler, SolidWorks98Plus. In exchange for trying out the new modeler, they would receive a complementary SolidWorks t-shirt. Within a period of two weeks after the initial session modeling the valve body, seven students had a session with SolidWorks98Plus. It was confirmed at the session that they had never used or seen demonstrations of the software before.

When the students arrived for their session with SolidWorks, they were told that they would be modeling the valve body again. In addition, during this session, both their actions on the computer and any conversation would be recorded on videotape. This was done with a freestanding microphone and a scan converter connecting the monitor to a video recording deck.

The session was run as a 'question-asking' protocol (Kato, 1986; Mack & Robinson, 1992). That is, it was acknowledged that they had never used the software before, but they were to try – to the best of their ability – to figure out how to model the part. At the same time, if they had any questions about how to use the software, they were to ask the investigator. The investigator, in turn, would only attempt to directly answer the questions raised by the students and not provide any additional instruction.

- **Interface Search.** Questions concerning where to find a command that performed a certain type of action, how to use a particular command, or a query about the result of an action (e.g., "How do I create a new datum plane?").

- **Errors.** Questions or comments concerning an unexpected or confusing result of a previous action. It could also be recognition of an error in strategy (e.g., "Why did I get this error when I tried to pattern [the tab features]?").

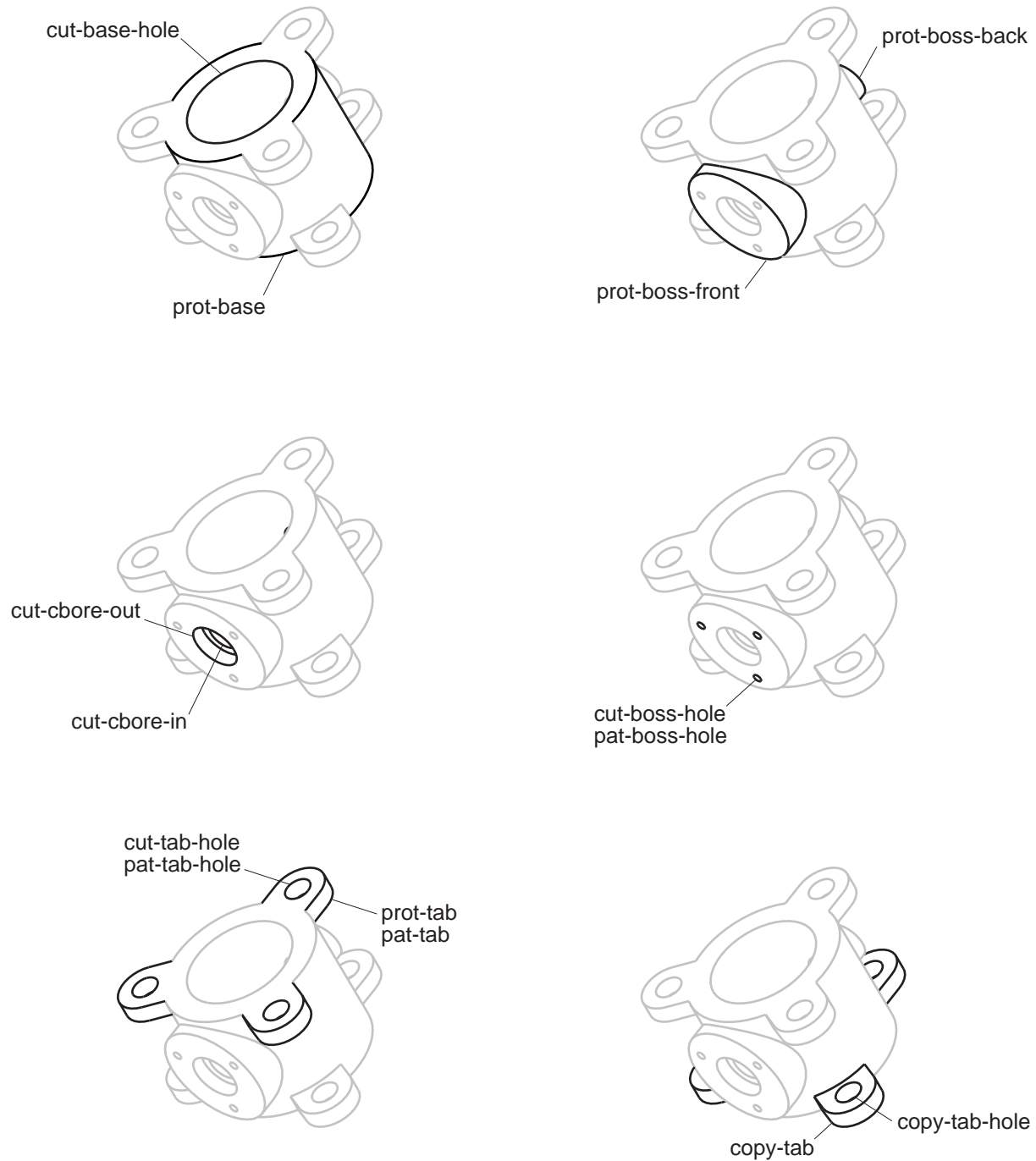


Figure 2 - Feature Operations

Results

Total modeling time

Figure 3 shows the modeling time taken to model the valve body in Pro/E and SolidWorks. The time recording began when the introduction was finished and stopped after the last feature operation was completed. The general trend indicates longer modeling time in Pro/E (between 0:45 and 2:28) than in SolidWorks (between 0:30 and 0:44).

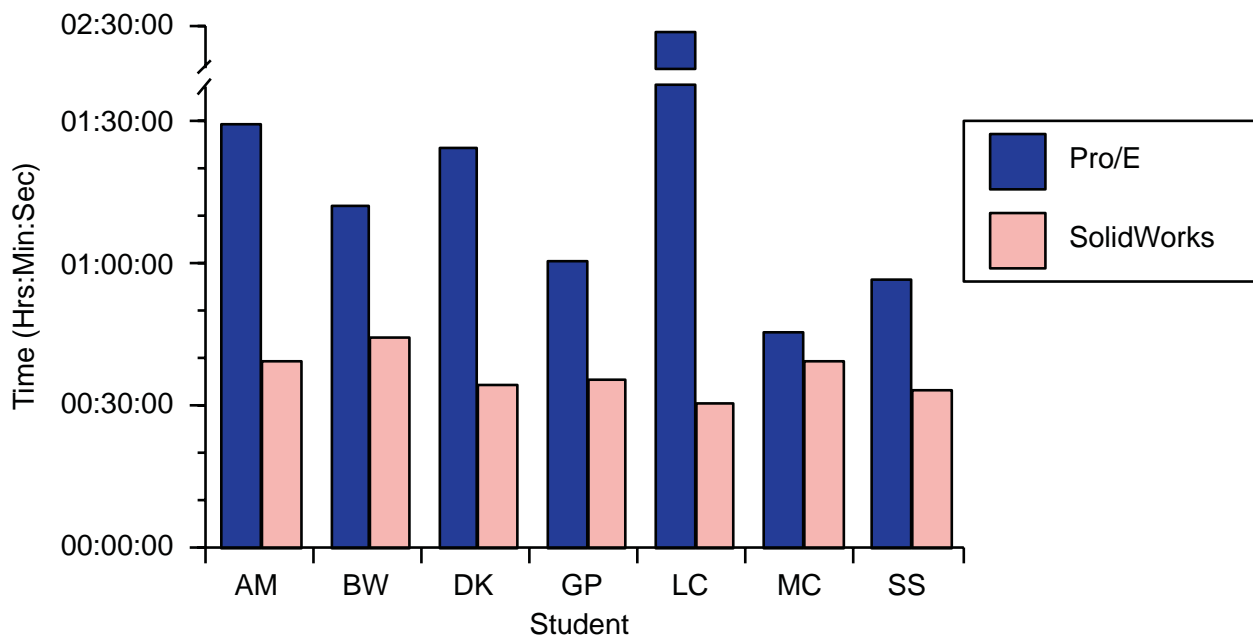


Figure 3 – Total modeling time

Operation sequences

Though not enough students were tested to be able to generate statistics based on probabilities, descriptive statistics do give an idea concerning some of the trends that emerged when examining the sequence of model feature operations used to create the valve body in both Pro/E and SolidWorks. The scatterplot shown in Figure 4 indicates that there were no clear trends concerning the number of operations used by any individual student or by the student group as a whole across software packages. The number of feature operations used in Pro/E ranged from 10 to 15 while the number of operations used in SolidWorks ranged from 11 to 15. The lines connecting points indicate the change in the number of operations for each student between packages. One student used the same number of operations while four used more operations in SolidWorks and two used fewer operations in SolidWorks.

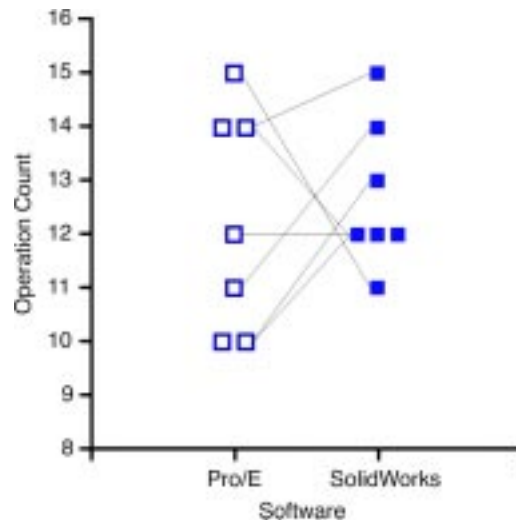


Figure 4 – Feature Operation Counts

Also of interest is the actual sequence of feature operations used by the student. There were no overall feature sequence trends recognizable which spanned the entire modeling process. Though all students (with both packages) started with prot-base (see Figure 2), there was no trend as to what the second operation was. There was also no consistency either within students (across packages) or across students concerning whether they combined cut-base-hole with prot-base by including a second profile loop. Two students combined them with Pro/E and two with SolidWorks, but no student did it with both packages. Concerning the combining prot-tab with cut-tab-hole, four students did this with Pro/E and three with SolidWorks with three out of the four doing it with both packages. What was also seen were local trends of pairs of operations. Boss-cbore-in and boss-cbore-out were paired together in 13 out of 14 instances (7 students x 2 packages). In creating the bosses, boss-front was always done before boss-back (14 out of 14 instances), though boss-back was not always the next operation. Similarly, when a feature or group of features was to be patterned or copied, this typically happened directly after the initial feature was created. This happened 13 out of 14 instances for pat-tab+hole, in 13 out of 14 instances for copying the tabs from the top to the bottom of the body, and in 14 out of 14 instances for pat-boss-hole.

The actual geometry created by the feature operations were very consistent between both packages (and within students) and between students. Where the geometry represented in Figure 2 did vary, it tended to be from:

- Whether the hole in the base and in the tab were combined into the protrusions or done as separate cut/hole operations.
- Whether copying the tabs from the top to the bottom of the base was done with a pattern, general copy, or mirror copy command.

In addition, where the model features were created relative to datums also varied. For example, though all students created the initial base protrusion so its central axis intersected with two out of the three initial datums, in some cases the third datum was located at a flat end of the cylinder while at other times was located bisecting the cylinder. Whether any datums were created after the initial three depended, in part, on when the base protrusion was cut out. If this was done early in the modeling process, then datums had to be located at the flat faces

of the two bosses with the bosses extruded back to the surface of the base. If the base cut came after the bosses, then the bosses could be extruded from the center of the base.

Transcribed dialogue

The transcribed question-asking protocol provided considerable qualitative information. Three categories by which statements were categorized were Strategy, Interface Search, and Errors (See definitions in the previous section). Table 1 shows a total of 551 transcribed statements fell under these categories. It should be noted that for Strategy, these were typically singular statements made by the students, while dialogue under Interface Search almost exclusively came in diads: a question by the student and a response by the investigator. This makes the total number of interface search topics roughly half the number shown in Table 1. The Error category was a mixture, since some dialogue was simply statements of error while others asked questions that received answers from the investigator. Still, these totals give an idea to the range of the quantity of dialogue that was recorded both between categories and between students.

| Student | Errors | Interface Search | Strategy | Total |
|--------------|--------|------------------|----------|-------|
| AM | 39 | 58 | 22 | 119 |
| BW | 19 | 40 | 17 | 76 |
| DK | 7 | 42 | 18 | 67 |
| GP | 42 | 42 | 27 | 111 |
| LC | 14 | 50 | 5 | 69 |
| MC | 19 | 18 | 2 | 39 |
| SS | 11 | 45 | 14 | 70 |
| Total | 151 | 295 | 105 | 551 |

Table 1 – Counts of transcribed statements by type

Within the category of Strategy, coding subcategories were defined as following (number of statements listed afterward):

- Viewing 11
- Feature Choice 30
- Sketch Plane Creation ... 13
- Sketch Plane Selection .. 22
- Sketch Creation 6
- Sweep Definition 19
- Other Selection 4

These categories are parsimonious with the generic modeling strategy outlined in Wiebe (1999). The most common statements fell under the category of Feature Choice (30) followed by Sketch Plane Selection (22) and Sketch Plane Creation (13). Only six statements were coded under Sketch Creation with all of these statements generated by just two of the students. Below are examples of different kinds of strategy statements:

Feature Choice

BW: “So now I’m interested in mirroring the whole bunch [of tabs] down....”

Sketch Plane Selection

GP: "What datum could I use.....Oh I could use that one [Datum 1] coming from that [back] side..."

Sketch Plane Statement

SS: "Wait, I just remembered the other time [in Pro/E] I did this, I set a plane out here [outside of prot-base]."

Within the category of Interface Search, coding subcategories were defined as following:

- View/Show44
- Select Feature41
- Draw6
- Dimensional Constraint21
- Modify Dimension Value27
- Pattern/Copy24
- Operation Specification72
- Geometric Constraint44
- Datum Creation8
- Feature Deletion8

The most common question category was Operation Specification (72), with Geometric Constraint (44), View/Show (44), and Select Feature (41) all about equal in the next level. Datum Creation (8) was only coded for two students, while Feature Delete (8) and Draw (6) was only coded for three students. Below are examples of different kinds of strategy statements:

Operation Specification

LC: "How do I know if it is going to extrude both sides?"

Investigator: [explains selection in dialogue box]

Geometric Constraint

AM: "Will the new circle I draw be centered with the circle [prot-boss-front]?"

Investigator: [Explains how to interpret dynamic icons on cursor that change as you pass over geometry]

Select Feature

BW: "Did it assume that I wanted that plane [to create the sketch on]?"

Investigator: It was already highlighted when you picked the sketch tool.

Within the category of Errors, coding subcategories were defined as following:

- Sweep Direction 24
- Sketch Location 17
- Feature Operation 7
- Operation Specification . 58
- Modify-Regenerate 16
- Select Feature 17
- Feature Delete 6
- Thin-Solid 6

The most common interface error was Feature Operation (58) with Sweep Direction (24) a distant second. These were followed by Sketch Location (17). Only one student did not have at least one Sketch Location error. Only four of the seven students had Modify-Regenerate errors (16).

Below are examples of different kinds of strategy statements:

Operation Specification

MC: [gets error message after trying to copy] "How do I find the direction that the copy is made in?...Ahh, I need to give it a face [sic] to go to just like I used a face [sic] to go around before [when I did a radial pattern]"

Investigator: "Pick an axis to give a vector for the direction."

Sweep Direction

GP: [Gets error about disjoint body] "Do you know what this means?"

Investigator: "You probably extruded in the wrong direction."

Sketch Location

GP: "Oh, I'm on the wrong datum....I'm going to make a datum out here [at outside surface of the front boss] and extrude back to the surface [of the base]"

Error codes were also searched for in conjunction with other codes to try and find commonalties between errors and particular activities. For example, the Operation Specification error code was intersected with the pat-tab feature operation to find that five out of seven students had errors in specifying this operation. A sample of the type of problem they were having is given above in the Operation Specification example. Of these five, four tried this pattern operation before they tried pat-boss-hole. The fifth student had errors on both pat-boss and pat-tab. Of the two students who didn't have errors on pat-tab, one did pat-boss-hole first and had errors with it. The copy-tab operation also coincided with specification errors for three students. This operation was most commonly done with a linear pattern command. No other cut or protrusion operation coincided with an error in more than one student.

The Sketch Location error code also coincided with the prot-boss-front operation for three of the seven students. An example of this dialogue:

BW: "Here I might be in trouble for drawing the boss..."

Investigator: "The reason being..."

BW: I don't have a datum plane [I can use]..."

On the other hand, there were no instances of Sketch Location errors coinciding with prot-boss-back even though it is a very similar feature. It is interesting to note that all students created prot-boss-front before prot-boss-back.

Discussion

The goal of this study was two-fold: first, to demonstrate some techniques which have not been widely used in the development of instructional materials, and secondly, to look specifically at issues pertaining to transfer of training from one 3-D constraint-based modeling package to another. In this study, transfer of training was used as a vehicle to get students to reveal elements of their knowledge. Using Shneiderman's (1998) OAI model, this knowledge was roughly divided into two areas: specific, syntactic knowledge of the interface of constraint-based modelers and more general, semantic knowledge of constraint-based modeling. Using a 'question-asking' protocol, the investigator was able to elicit information from the students that helped elaborate on their syntactic and semantic knowledge of constraint-based modeling software. In addition, the sequence of model operations in Pro/E and SolidWorks could be analyzed in its own right and combined with the protocol data to reveal more information about the process.

The results of the study clearly show that students were able to transfer knowledge which they learned from working on Pro/E for most of a semester and

use it to create a model in SolidWorks. Using Schumacher & Czerwinski's (1992) definition, the student's understanding of 3-D constraint-based modeling was clearly moving towards the 'expert' end of the spectrum by the point in the semester that this study was run. Understanding of Pro/E went beyond memorization of surface features (e.g., color/shape of an icon or location of a command) of the software. The student's understanding had enough depth to be able to generalize about the functionality of the system and how it might be used to fulfill task goals. This abstracted knowledge clearly revealed itself when the students sat down and used SolidWorks for the first time.

Though it is important not to read too much into the total modeling time results, it is clear that a combination of factors allowed students to model the valve body more quickly on SolidWorks than in Pro/E. Likely factors include:

- positive transfer of training of modeling knowledge
- the general usability of the SolidWorks interface compared to Pro/E
- the ability to have one-to-one support of the investigator when modeling in SolidWorks

Addressing the first point, it is likely that there was positive transfer of a general strategy as to how to model the part. Very few utterances by the students were asking for support as to how to model the part. The statements on Strategy and questions concerning Interface Search clearly indicated they had a plan of action in mind. This plan of action was quite parsimonious with the generic modeling strategy proposed by Wiebe (1999). This generic modeling strategy served as a robust structure for coding the modeling strategies, the questions concerning commands on the interface, and errors that they might have committed. The breakdown of the geometry into features showed a high degree of similarity between the parts modeled in Pro/E and those modeled in SolidWorks. In addition, though there was no discernible overall sequence of modeled features, more local feature operation sequences revealed themselves in similar patterns in both modeling software packages.

Though a general modeling strategy could be carried between the two software packages, there was a clear learning curve for mastering the syntactic elements of the new software, SolidWorks. Though the basic modeling elements of defining a profile on a 2-D plane and sweeping it using a feature operation was the same, there was a clear difference in order in SolidWorks. None of the students were able to create the first feature without being instructed that you first create the sketched profile and then chose a feature operation to match it. Another element different in SolidWorks was the clustering of most of the feature operation specifications (e.g., direction of sweep, depth of sweep, etc.) in a single dialogue box and the specifics of how they operated. A clear negative transfer of training example which appeared with a number of students in this dialogue was the fact that a two sided sweep did not, by default, go an equal distance on both sides of the sketch plane as it did in Pro/E.

There were a number of cases where tasks were automated, implied, or not needed to be completed in SolidWorks while in Pro/E they were explicitly required to complete a particular task. For example, in SolidWorks the automatic updating of a sketch profile when a dimensional constraint was modified caught some students off guard initially. In fact some students still requested to know where the regeneration command was and, when they used it, it popped them out of the sketcher, creating new problems. Similarly, when using the patterning command, students were often initially at a loss as to how to use the dialogue box

rather than being stepped through a series of parameter definitions. Some did not initially specify an axis for radial patterning and were given an error message when they tried to complete the operation. At the same time, it usually did not take more than one try at a command before the new syntax could be integrated into their higher-level modeling strategy. In all but one case, if a student made an error the first time with the pattern command, they did not the second time through. Though students all needed help on creating prot-base (the first feature for all), they rarely created errors in the basic mechanics of creating further protrusions and cuts using simple circular profiles.

Though a majority of the errors committed were associated with learning the syntax of the new modeler, there were also still some higher level strategy errors being made. Even though the students had previously modeled this part less than two weeks prior to the session with SolidWorks, there were still strategy errors in feature operation sequence being made. An example of this was planning how the bosses were to be made based on whether cut-base-hole was already performed. Though it was not possible to evaluate whether similar strategy errors were made in Pro/E, they did occur in SolidWorks the second time modeling the part. It could be that the effort of learning a new software package drained focus from developing a long range modeling strategy.

The results of this study can provide some guidance when developing appropriate instructional strategies for teaching 3-D constraint-based modeling. This study gives some insight in to how easy it can be to transfer higher level modeling strategies between software packages. Though there were obvious hurdles to overcome in learning a new interface, with a solid strategic structure in place, this new syntax seemed to be readily absorbed into the task strategy. These lessons not only hold for students as they move out into the workplace, but also apply knowledge they have acquired in class to other classes and to the same class later in the semester.

There is support from other researchers that an educational model for such a course should make use of an instructional strategy which maps well to both specific syntaxes of specific software tools, but can also be generalized easily to other packages (Waern, 1993). This may argue for less depth in learning absolutely every command available in a particular package (i.e., 'knowing it cold'), and promoting more breadth spanning software packages (even if you aren't teaching them all specifically) by developing educational models which emphasize a higher level understanding of the commonalties between systems. This higher level understanding will still come in the context of direct, hands-on practice with a modeling package. It is here that the student can test the hypotheses generated from his or her mental model and revise the model as appropriate.

Less time spent on learning all of the specific syntax of a software package can also mean more time spent on activities that emphasize problem solving using constraint-based modeling. As students get to know the basic functionality of the constraint-based modeler, it is errors in higher level strategy that tend to cause a larger proportion of the problems. Exercises which reward higher level thinking rather than encyclopedic knowledge of a specific software tool will be of longer term utility to the student. Putting these exercises in the context of constraint-based modeling will also allow the students to further develop appropriate mappings of engineering and design tasks to the generic capabilities of constraint-based modeling.

Further Work

Though this study certainly pointed to the utility of many of the research techniques used, it also points to weakness of their particular application here. For example, more detailed coding was needed of the Interface Search questions, especially understanding the different types of operation specification problems students were having. This may lead to a finer understanding of the transfer of training issues involved in moving between packages. Also, a better understanding is needed of the relationship of the strategy statements and the interface search questions. In many cases the strategy statements either were located sequentially with interface search questions or overlapped them. One possibility is merging of strategy codes with interface search questions and seeing what patterns emerge.

With more subjects, tests can be run to calculate the statistical probabilities of local sequences of operations. With just seven students, certain patterns seem to be emerging but could not be rigorously evaluated. First order Markov chain analyses can test for occurrences of one operation directly after another, while lag sequential analyses can be used to see the probabilities of an operation occurring two or more steps before or after another one (c.f. van Hooff, 1982). A better understanding of sequences of operations can help explain strategies students (or professionals) apply in developing a model.

One logical way in which this work could be expanded is to look at different combinations of software packages in order to see if the same syntax issues emerge. Similarly, the same software packages could be used, but different modeling problems could be used which highlight different strategies (or the failure to fully develop one). Another approach would be to use students at varying levels of experience with their initial package before trying the new package. All of these approaches would expand on the understanding of how students form a mental model of how constraint-based modelers function and how to apply them to engineering and design tasks.

References

- Anderson, J. R. (1982). Acquisition of cognitive skill. Psychological Review, 89, 369-406.
- Arnold, E., & Bessant, J. (1983). Oiling the wheels of technical change: Skills, training and the adoption of computer aided design. In J. Rijnsdorp (Ed.), Training for tomorrow: Evaluational aspects of computerized automation (IFAC/IFIP Conference), (pp. 123-127). Amsterdam: Elsevier.
- Baldwin, T. T., & Ford, J. K. (1988). Transfer of training: A review and directions for future research. Personnel Psychology, 41, 63-101.
- Bertoline, G. R., Wiebe, E. N., Miller, C., & Mohler, J. L. (1997). Technical graphics communications. (2 ed.). New York, NY: McGraw-Hill.
- Carroll, J. M., & Olson, J. R. (1990). Mental models in human-computer interaction. In M. Hellander (Ed.), Handbook of human-computer interaction, (pp. 45-65). Amsterdam: North-Holland.
- Kato, T. (1986). What "question asking protocols" can say about the user interface. International Journal of Man-Machine Studies, 25, 659-673.
- Klein, K. J., Hall, R. J., & Laliberte, M. (1990). Training and the organizational consequences of technological change: A case study of computer-aided design and drafting. In U. E. Gattiker & L. Larwood (Eds.), End-user training. Berlin: Walter de Gruyter.
- Mack, R., & Robinson, J. B. (1992). When novices elicit knowledge: Question asking in designing, evaluating, and learning to use software. In R. R. Hoffman (Ed.), The psychology of expertise: Cognitive research and empirical AI, (pp. 245-268). New York, NY: Springer-Verlag.
- Schumacher, R. M., & Czerwinski, M. P. (1992). Mental models and the acquisition of expert knowledge. In R. R. Hoffman (Ed.), The psychology of expertise: Cognitive research and empirical AI, (pp. 61-79). New York, NY: Springer-Verlag.
- Shneiderman, B. (1998). Designing the user interface: Strategies for effective human-computer interaction. (3rd ed.). Reading, MA: Addison-Wesley.
- van Hooff, J. A. R. A. M. (1982). Categories and sequences of behavior: Methods of description and analysis. In K. R. Scherer & P. Ekman (Eds.), Handbook of methods in nonverbal behavior research, (pp. 362-439). Cambridge, UK: Cambridge U. Press.
- Waern, Y. (1993). Varieties of learning to use computer tools. Computers in Human Behavior, 9(2-3), 323-339.
- Wiebe, E. N. (1999). 3-D constraint-based modeling: Finding common themes. Paper presented at the ASEE Engineering Design Graphics Division Mid-Year Meeting, Columbus, OH.