Exploring the Process of Adult Computer Software Training Using Andragogy, Situated Cognition, and a Minimalist Approach

Andrew C. Hurt
Texas A&M University

With technology advances, computer software becomes increasingly difficult to learn. Adults often rely on software training to keep abreast of these changes. Instructor-led software training is frequently used to teach adults new software skills; however there is limited research regarding the best practices in adult computer software training. This paper looks at how software trainers incorporate the adult learning theories of andragogy, situated cognition, and the minimalist training approach into adult computer software training.

Keywords: Software Training, Adult Learning, Situated Cognition

For the novice adult learner, computer software can be difficult, troubling, and often impossible to understand. When faced with learning a new software program, adults generally seek help in learning how to use the software (Gupta & Ndahi, 2002). Often this help comes in the form of instructor-led computer software training. Additionally, with our society's ever increasing reliance on computers and the changing nature of software, adults often find it difficult to keep up. The U.S. Department of Labor projects that between 2004 and 2014 employment in "management, scientific, and technical consulting services also will grow… by 60.5 percent, spurred by the increased use of new technology and computer software and the growing complexity of business" (Tomorrow's jobs, 2003). Thus as technology becomes further integrated into society, there becomes increased demand for software training geared to the novice adult learner.

Although there is an abundance of training literature, there is limited research that looks at successful strategies to train adults specifically in computer software. A simple literature search conducted on the term "training" in four databases over the last ten years (1996-2006) results in over one hundred thousand articles. A search using the same databases and search parameters using the term "computer software training" yielded slightly over 700 results. Finally, when those results are further siphoned through the term "adult" or "andragogy" the list decreases to approximately 100 articles. Of those 100 results, very few have anything to do with the process of software training specifically geared toward adults. Thus the problem is that there is a growing need for increased understanding on the adult software training process and at the same time a deficiency in the literature regarding this topic

The purpose of this study is to explore how trainers successfully incorporate the adult learning theories of andragogy and situated cognition with the minimalist approach to software training into the software training of adults. In order to accomplish this task this study is guided by four research questions: (a) How is the minimalist approach to software training is being used? (b) How do trainers present the material in a manner which facilitates the adult learning theory of situated cognition? (c) What evidence is there in the training environment of the five principles of andragogy? (d) In regards to the process of computer software training, is there a connection between the minimalist approach, situated cognition, and andragogy? In this paper a discussion of the literature as it pertains to adult computer software training is presented, the results of a qualitative study focused on identifying these successful practices is outlined, and implications to the field of Human Resource Development (HRD) are discussed.

Literature Review

When discussing the training of adults, this study would be severely lacking if it did not discuss the theory of andragogy. Proposed by Malcolm Knowles (1968) andragogy is the prominent theory of adult education. Taking its roots from pedagogy (theory of adolescent learning); Knowles suggested that adults learn differently than adolescents. He made five assumptions of adult learners. The first assumption is that as a person matures they move from a dependent personality (adolescent) towards a more self-directed personality (adult). This self-directed personality is used to seek out information in order to gain new skills and competencies. The second assumption is that adults have an ever growing pool of experience. When new learning occurs, adults tie this new learning to their prior experience. The third assumption is that the readiness of adults to learn is tied to their social role. The fourth

assumption is that as humans mature they move from a need to have immediate application knowledge to more future oriented problem solving skills. Adult learners want problem solving skills not just immediate answers. The final assumption is that adults are motivated to learn by internal factors not by external ones. Learning motivation comes from an internal desire to understand the problem, not by external forces pushing adults to learn.

Tied to the way that adults learn is the situation in which learning occurs. Situated learning or situated cognition is an adult learning theory that says that learning and the situation are so closely related that they can not be separated (Merriam & Caffarella, 1999). According to Hansman (2001), "the core idea of situated cognition is that learning is inherently social in nature" (p. 45). Situated cognition's inherent social nature means that the activity learned, tools used, interaction among learners, and the context in which the learning takes place all play a factor in successful integration of new knowledge.

Traditionally, learning has been viewed as theoretically independent of the situation (Wilson, 1993). From a situated cognition perspective, new knowledge is extremely difficult to obtain without learning the material in the context in which that material will be used. Wilson provides numerous examples of studies that have shown transfer of learning to be context based. Although there is some disagreement in the adult learning community, "proponents of a view of cognition as inherently situated offer evidence that adult learning and knowing are profoundly structured by the context in which they occur" (p. 74).

Michal Hughes (1998) suggested that to meet the needs of adult computer software learners, trainers need to approach the training instructional design process from a situated learning point of view with an emphasis on the principles of andragogy. In his article Hughes lists several points that need to be considered when designing training around this topic. First, the situation in which training will be conducted is integral to the acquisition of knowledge. Second, trainers must provide the necessary "scaffolding." Scaffolding is the support network that is provided to the student. Third, there needs to be supports in place that help the instructor. These supports might track student progress, gauge understanding, or help the instructor interact with students. The final point is that there needs to be a solid definition of the role and nature of assessment in the training. This definition will help distinguish the boundaries by which students and the instructor can work.

Of the computer software training literature there is little empirical research that specifically focuses on adults. Several studies have been conducted on undergraduate college students; however, it is difficult to gauge whether these individuals would be classified as adults. Based on Knowles (1968) assumptions of andragogy, there is no definitive way of determining an exact age at which an adolescent becomes an adult. An undergraduate college student is typically 18-22 years old, thus the problem is that this is often the age at which individuals in U.S. culture often transverse between adolescents and adulthood. Therefore, findings from studies of undergraduates may not have transferability to work related contexts.

There are a few studies that explore software training involving adults. Harp, Snadra, and Satzinger (1998) looked at the method that was used to teach computer software training. The focus of the study was to identify differences in perceived usefulness of three forms of software training. The first form was computer-based training (CBT); which involves participants using instructional software to learn how to use other forms of computer software. The second form of training was a video tutorial. In this form, participants watch a video which instructs them on how to use the software and then participants follow along on their computers. The third form was instructor-lead classroom training. In this final form, participants learn to use software by watching and listening to an instructor. Then the students complete the task on their own computers. Results of the study by Harp et al. indicated that users preferred the CBT and instructor-led training over the video training. No difference was found in the preference between CBT and instructor-led training.

Similar to the method used in software training, Judith Lambrecht (1999) looked at the overall approach to software training. She suggested that when conducting software training there are two avenues that an instructor could use. The first is a traditional systematic approach. In this approach the instructor explicitly explaining what to do to solve a problem. One advantage of this approach is that it teaches the learner exactly what to do in a specific situation. A disadvantage is that when faced with situations that they have not experienced, learners have difficulty finding solutions to problems. The second avenue for teaching software training is the minimalist approach. In this approach instructors teach learners basic problem solving skills. The minimalist approach was developed by John Carroll in the 1980s (Carroll, 1990). Carroll suggested that software training is a "paradox of learning…to learn, the student must interact with the system; but to interact with the system, the student must learn"(p. 147). Hence one advantage of the minimalist approach is that when faced with new problems learners apply their problem solving skills and find the correct solution on their own. Instead of the systematic approach which only teaches students to solve problems that they have previously encountered.

Lambrecht (2000) then tested the systematic and minimalist approaches to software training in a year long qualitative study. She conducted 46 interviews in total, across three "exemplary" high schools in the state of

Minnesota. Interviews were conducted with a mix of students, teachers, and staff. Her results showed that the dominant method was the systematic approach. Lambrecht's results appear to be consistent with the tenets of systematic training considering that the systematic approach is characterized by a high learner need for dependency. Although Lambrecht (2000), did not exclusively use adult learners, considering Knowles's first assumption of andragogy that adolescents are dependent in regards to their learning. Lambrecht's results could suggest that had she conducted her study in an adult oriented environment instead of a high school, she likely would have found that the minimalist approach would have been the dominate training method. This is because as andragogy assumes, "as a person matures they become more self-directed" (Merriam & Caffarella, 1999, p. 272). Having reviewed the dominant literature regarding the adult learning theories of andragogy and situated cognition and having looked at the literature regarding the process of the minimalist approach to training, it seems appropriate to explore how trainers incorporate these three principles in the software training of adults.

Methodology

This study took a grounded theory approach to discovering how software trainers use the minimalist approach, situated learning, and andragogy in the software training of adults. Grounded theory is a type of qualitative methodology where in the researcher is attempting to develop theory from the data that is presented (Merriam, 1998). Grounded theory is a process of inductively discovering the phenomenon in question (Egan, 2002). In this study software trainers were interviewed and then based on those results, the process of incorporating the minimalist approach, situated cognition, and andragogy into the software training of adults was derived.

Prospective Study Participants

In order to explore this process on how software trainers incorporate the minimalist approach, andragogy, and situated cognition into the training of adults. The experiences of two software trainers employed in the training department of an organization located in the southern part United States were analyzed. These two individuals were chosen because they exemplify the "best" the department has to offer in regards to software trainers. In this study the participant's names have been changed in order to protect their anonymity.

The first trainer, named Jerry, is a middle-aged white male. Jerry has been conducting software training for the past 11 years. The last six of those years have been as an employee of the organization used in this study. Jerry's prior experience includes working as a debit card trainer and providing software training for his family's business. Jerry primarily focuses on teaching Microsoft Office courses (Word, Excel, Frontpage, Publisher, Outlook), Adobe courses, and introduction to Windows/computers courses.

Matt, the second trainer that was interviewed is a middle-aged white male. Matt is a graphic artist by training and as a result got into using the first version of Powerpoint. Matt has been teaching various software packages for the last 8 years. Although, he has only been employed in the organization for one year, his experience is extensive. Matt primarily teaches the Microsoft software; Powerpoint, Publisher, Access, Project, Visio, and Frontpage.

Data Collection

The two individuals listed above were informally asked to participate in approximately an hour long interview. The interviews were conducted on the same day (one in the morning and one in afternoon) and both were recorded using a digital recorder. The purpose of this project was explained to both interviewees prior to and at the beginning of the interviews. The interviews were conducted using Patton's (1988) general interview guide approach. In this approach, the interviewer discusses the topic(s) of interest with the interviewee prior to conducting the interview. Then during the interview, the interviewer works those questions into the dialogue; rephrasing them as the situation/dialogue dictates. In general this is an approach used to discovery new spontaneous information, yet allows for the specific research questions to be addressed.

Data Analysis

After completing the interviews, the digital record of each was transcribed. A constant comparative method of coding the data was used in the analysis of the data (Coffey & Atkinson, 1996). As Merriam (1998) suggested, the constant comparative method involves comparing each individual piece of data with all of the others. In so doing, categories can begin to form around a specific topic. Central to the grounded theory approach; these topics were used as a basis for developing the emergent theory of how software trainers use andragogy, situated learning, and a minimalist approach in the training of adults on computer software. It should be noted that one of the tenets of grounded theory is that it is an ongoing process (Egan, 2002). Grounded theory involves continuously collecting data then analyzing. This process is repeated until commonality within the data is seen. The information presented here is merely the first stage of a multistage on going process. As more data is collected the results will further be clarified as the theory emerges.

Reliability and Validity

In order to control for validity and reliability, I employed as several authors (Peshkin, 1988; Wolcott, 1994) suggest a *rigorous subjectivity* and as Merriam (1998) suggests triangulation and member checks. I have strived to present my opinion and position as clearly as I can, while conducting the interviews, coding and analyzing the data, and writing this paper. As Wolcott suggests, clearly presenting the authors subjectivity is essential in creating a valid study because it allows the reader to make an informed decision about the author. Additionally, I have employed the triangulation technique which involves analyzing the data from as many perspectives as possible. Using the multiple perspectives allows the researcher to "triangulate" the truth of the data (Mathison, 1988). Finally, member checks have been employed throughout this study. Member check is a process where the researcher continually brings the data back to the subjects to make sure they are interpreting it correctly. After the data was transcribed, I provided my two interviewees with copies of the transcriptions and after the data was coded I went back again to my two interviewees and asked if they thought the information an accurate representation of what they told me in the interview.

Researcher Bias

All research and for that matter researchers are not without their own personal biases. The single biggest bias that I bring to this study is that I am a software trainer employed by the organization used in this study. In many ways this is a benefit, as a software trainer I have first hand knowledge of the work activities, cliental, and most importantly trainer abilities. However, there are some significant concerns that should be noted. First, being a software trainer, I have developed my own style of training. My personal style could influence the results of this study because I might inadvertently apply my perceptions about software training to the interview process and data analysis. This is a problem because then the results would be a reflection of my beliefs and not the interviewees. Second, I am employed by the organization used in this study. As with most organizations, there are rules and regulations that must be followed when running a training class. These rules and regulations could present a bias as they influence how I teach and for that matter how both Jerry and Matt teach.

Research Findings

This section presents the results of the two interviews. After the data was collected and analyzed five major themes emerged from the data. These five themes are presented in a graphical representation which can be seen in Figure 1. The five themes are: pre-training activities, systematic training activities, minimalist training activities, situated training activities, and andragogy.

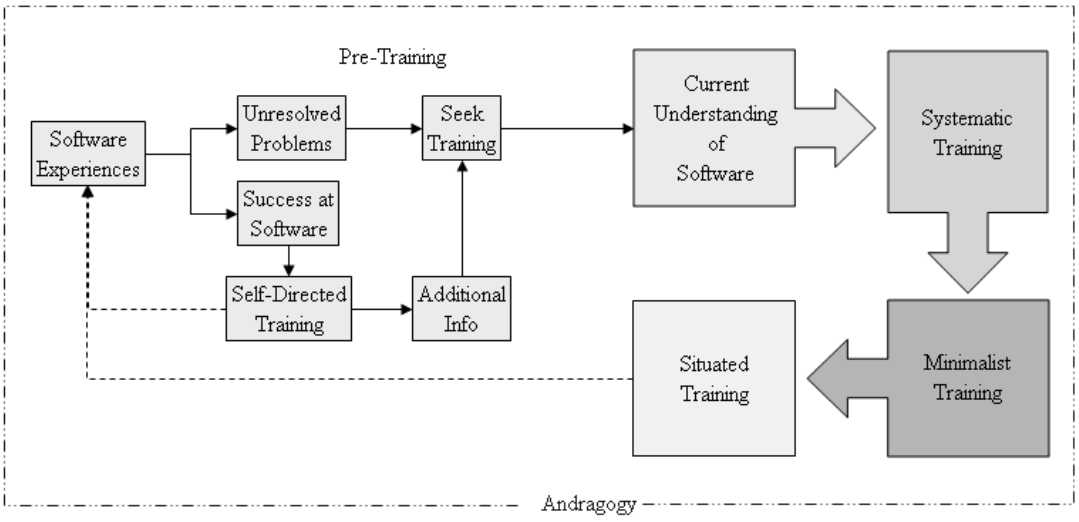

Figure 1. Training Process

Pre-Training

The first category which I have entitled pre-training consists of all the information the student currently has about the software. This category is depicted in Figure 1 and consists of the entire left half of the model. The pre-training category is unique in that the software trainer does not have the ability to change what the learner's prior

experiences are, but all of these experiences affect the way in which the trainer must structure the course to the learner. This process begins with the background that the student has with the software and with computers in general.

The student is then forced to make a choice. One way or another the student must learn the software program, so their choice becomes either to seek out a formal training solution or try and teach themselves the program. If the student has been successful in teaching themselves other programs then they will like attempt to teach themselves the new program. As Matt suggested, many times students will "look for online materials and practice as best they can with the software application."

However, if the student has not been successful at learning a new program in the past or they have unresolved issues that they can not seem to clarify on their own, then formal training is often sought. These unresolved problems do not develop because of the learner's inability to grasp the material, but rather as a need for support. As Jerry described it, "It's not that they can't learn the software it just that they have a fear of using the software. So sometimes you just have to hold their hand until they get more comfortable with it." Matt provides another example of a situation that drove the learner to seek formal software training:

> I once taught an 80 year old lady how to use a computer. When we first started she didn't even own a computer… she just wanted to learn so that she could email her grand kids and store/share recipes with her friends. It's a real testament to the desire of an individual to learn something new.

In Matt's example it was not that the grandmother was unable to learn the software, rather it was a function of her being intimidated by the software and thus needing a supporting hand. The desire of individuals to learn a new software program is not something that the instructor has the power to change, but these desires in part motivate and encourage the student to seek a formal training solution. This whole process of pre-training is important because it becomes the basis for what the trainer does have the ability to change in a classroom environment.

The last step in the pre-training category entails the student's current understanding of the software. This step also becomes the first of four areas that the instructor has control over. This step begins with the student attending the training session. It then becomes the instructor's job to try and ferret out the background knowledge that the student has. As Matt says, "I have to learn who my students are, what they want, and why they need this."

In order to work with students throughout the training and figure out the students prior experiences, the software instructor needs several skills. When asked what kinds of skills a "good" software instructor should have, Jerry said that "[Instructors] need to have basic speaking skills… instructors should also have class management skills." Although these skills seem to be relatively straightforward classroom teaching skills, Matt suggests a more nurturing skill set. "Computer software trainers need; enthusiasm, flexibility, empathy, and a real desire to help people… you know it may sound silly, but a good software trainer needs to be able to be human."

Systematic Training

Once the instructor has gained some basic information on the background experiences of their students, they can start to conduct the actual training. The training process starts with systematic training. The systematic training process entails the instructor explicitly guiding the student through the material. Both instructors in this study used course manuals during training. These manuals provide a step-by-step or "systematic" guide on how to use various components of the software. As Jerry said, "Software training is really about hard skills because you have to tell them to point here and click there. The lessons are really step-by-step."

This step-by-step manner in which the instruction is presented is useful to the student because it gives them the solution to an exact problem or question. Jerry elaborates on this idea and expresses a concern with it:

> A lot of software trainers are just your techy types. They show you how to do something, which is fine for people that just want the answers, but it's not really effective for most. It's effective for teaching the application, but it's not effective if you want anyone to remember what you did. Most [students] want to remember what you did and not what you said.

Matt, agrees with Jerry that step-by-step training does not create an effective learning environment. "In many ways I let the class dictate how they want to learn the material, but there are some basic instructions that need to be given just so that everyone knows how to use the software."

Based on Jerry's and Matt's assessments of the software training environment, the process of systematic training seems to be a necessary evil. It does not create an effective learning environment, yet the instructor has to do it in order to get all of their students up to the same level. Matt mentioned a strategy that he uses to get students more interested in the process and to make it more relevant to their lives:

> People like handouts. So I try to pullout informative things [from the lessons] and expand on them because when they get back to their offices the examples we use in class aren't going to be helpful. They need more general topics that have a wide use.

Many of the handouts that Matt gives to his classes follow this systematic approach. Matt's handouts layout in explicit detail, how to perform a specific operation. Although, the handouts clearly define what to do, in Matt's quotation it can be seen that he is unconsciously moving into the minimalist and situated categories of training with the phases "general topic have a wide use" and "when they get back to their offices."

Minimalist Training

The third category which I have named minimalist training requires the instructor to go one step beyond the systematic method. In systematic training the information is presented, but the problem is that the learning environment that systematic training requires is not conducive to the learner. Hence, there becomes a need to reinforce the concepts that were presented in the systematic training and to do so in a manner which will help the student to remember.

Minimalist training involves the instructor presenting the material under the guise of problem solving. The goal of the minimalist approach is to give the students a set of problem solving skills that they can use to learn and fix problems with the software after they have left the training session. It was clearly evident in my discussion with both trainers that they use several strategies to try and accomplish this. For instance, Jerry uses what he calls the "play" to teach the students how they can problem solve on the software:

> Sometimes I just play with the software during class… [But the trick is] you have to have an idea of how you're going to play to make it look like your [actually] playing … I plan out the play. After the play I come back and relate it to the objectives.

What Jerry is doing with his "play" is actually demonstrating during the training session that students can attempt to solve their own software problems and discover new ideas without having to attend a formal training course. This concept aligns perfectly with the minimalist approach because it suggests that students are learning to identify and develop their own solutions. When I asked Jerry if this has ever back-fired for him he said, "Sure that happens all the time. But when my playing around fails the students learn how to solve problems."

Matt on the other hand has two different strategies that he uses to teach problem solving skills. The first focuses on having the students form teams. "In my [Microsoft] Access classes, I sometimes have the class develop a database together; this helps them to solve problems together." By learning from each other, students start to develop a set of problem solving skills based on the small group interaction. This would also like benefit students later on as the potential is there to develop lasting relationships with fellow class members. Matt's second strategy is to subtly assist students while they are practicing the lesson activities. "…I really try and watch the students to see if their not getting it. If they aren't then I sit down with them and try to help them work through their problems… by helping them discover where they went wrong."

Situated Training

The third category is entitled situated training. Up till this point the learner has been presented the basic knowledge they need to know in order to use the software (systematic training), then they are taught how to solve problems on their own outside the training session (minimalist training), finally they need to be taught how this software is going to be useful to them (situated training).

In situated training the trainer's sole purpose is to explain the software in terms that the learner can connect with. This manifests itself in relating the software program's applications and functioning to the learner's job. In this category it becomes essential that the instructor has a solid understanding of the learner's prior experiences. Jerry and Matt both elaborated on why this is so important. However, I think Matt said it best, "its funny sometimes how a student really won't get the topic, but the minute you put their data in it…it makes perfect sense… that's the key, just find terms they can understand." Situated training focuses on connecting the applications of the software to the student's job. Jerry told me about how he often tries to connect the software to the student's job:

> I often have students bring in an example of their work…, I once had a lady who had an entire years worth of information on one worksheet [in Excel]… it was a matter of showing her how to separate the information out… which really helped her.

In Jerry's examples he helps the student to discover the applicability of the software to their job. In doing this the student is better able to grasp the potential uses of the software, which further increases their desire to learn.

Similarly, when Matt is developing a class, he said "I try and develop scenarios in my head that are going to help the students in their jobs. Because that's what really matters, connecting the software back to them." These scenarios that Matt develops are his way of situating the training in the jobs of his students. Matt also relayed to me a story of how he connected the learning to the student:

> I had a woman who came to Access training and she was an Excel person [she did not think Access would be useful]. Now Excel and Access are completely different programs. So [during class] I would say you can do this in Excel, but here is where Access goes far beyond what Excel can do. As soon as I showed her how to link records she got on board with Access.

In this story, Matt interacts with a student in order to convince the student that Access is a useful program. To do this he situated the example in a way that showed her Access's applicability beyond Excel. The result was that the student saw a new way of looking at Access; one that would help her in her job.

Finally, situated training ends the training process. The learner has all of the knowledge necessary to use the software in a productive way. All of this knowledge then becomes apart of prior experience, hence the dashed line in Figure 1. The knowledge gained from the three types of training (systematic, minimalist, and situated) relates back to experience. Thus, in the student's mind the training course will serve as prior experience to be drawn from the next time software problems are encountered. However, the entire process has not been completely explained.

Andragogy

The final category is simply entitled andragogy. Andragogy plays a vital role in the software training process because andragogy mediates the entire relationship. In every step of the process the software trainer must be cognizant of the principles of andragogy. Evidence of this is clearly seen throughout the dialogue that has been presented thus far. Additionally, both trainers made other comments throughout the interviews which relate to andragogy. For instance as Matt puts it, "I've never really had a student who truly didn't want to learn. Just about everyone who comes to my classes has a desire to learn the material." As andragogy states, this quote demonstrates that adults come to learn. A final illustration of how software trainers think about androgogy's role during the entire process can be seen from Matt:

> I think about what kind of people I'm going to have in my classes. Take for instance Acrobat, I'll be teaching that [class] in a few weeks and I'm thinking that the students in my class will likely not be first time computer user, [likely they will be]… more professional individuals who are going to be using acrobat to produce formal documents and things like that.

In Matt's development of the Acrobat class, he is conducting a future oriented thinking and trying to identify what kind of students will be in his class. Andragogy assumes that adults have an ever growing pool of experience. It is this experience that Matt is considering as he prepares for his class. This category of andragogy ends the discussion of the findings section, however it should be note that andragogy plays an extremely important function to the entire process because of its role as a mediator. In essence a discussion of all the other roles can not ensure with out also discussing andragogy.

Summary & Discussion

This study focused on identifying how software trainers use the minimalist approach to training, situated cognition, and andragogy in the software instruction of adult. To understand how they do this, two software trainers employed in the training department of a large organization located in the southern United States were interviewed. The results of this study found that effective software training can be divided into five components. First there is the pre-training; these are all the experiences that learners bring with them into the training sessions. Second there is systematic training. In this type of training instructors present the material in a step-by-step fashion. The trainer explains every detail to the learner. Although useful for knowing how to use a software application this type of training is deficient in its ability to solve problems. Third, there is minimalist training; this type of training focuses on giving the learner problem solving skills. Problem solving skills help the learner by improving their ability to answer their own questions. Fourth there is situated training. This is the highest most detailed level of training. In situated training the instructor attempts to connect the learning back to the students work environment. This provides the learner with an understanding of how to use the software within their own work context. The final category is andragogy; of all the categories this may be the most important because it mediates the entire relationship described thus far. Both software trainers interviewed showed clear evidence that they used the principles of andragogy throughout the entire process.

Discussion

As was presented at the beginning of this paper; this research study was guided by four questions. As for the first question, there was clear evidence depicting how the minimalist approach to training was used. However, a concern with this finding is in regards to Lambrecht's (1999, 2000) discussion on when to use a minimalist approach. She suggested that trainers should uses either a systematic or a minimalist approach when conducting training. However, what these findings indicate is that both approaches are being used. The systematic is used at first to explain the basics of how the software works, and then the minimalist approach is used to further clarify and teach the students more specific problem solving skills.

The second question was also clearly addressed in this study. The highest level of training was found to be based on situated cognition. There were also many other aspects of this study that seemed to incorporate situated cognition. In fact the entire idea of using computer based training to teach people software is situated concept.

For the third question, andragogy was found to be the mediating variable across this entire process. Andragogy was always in the minds of the two trainers. Both considered their student's success foremost on the list of training objectives and in doing so this required them to think in andrgological terms. Andragogy is so important to this study that it is impossible to discuss one component without also discussing andragogy's effect on it. However, this is not an unexpected phenomenon considering the fact that Knowles developed andragogy to be a set of guiding principles in adult learning. In this study that is what andragogy was show to do; be a guiding principle in minds of software trainers.

Finally, to address the fourth question there is clearly a connection throughout the process between all three theories. In reviewing the results section, within each category there are elements of all three of these concepts. Admittedly, there are dominant themes within each category however there certainly are elements of each throughout the entire training process. Unfortunately, it is beyond the scope of this study to accurately explain how that relationship works. More detailed research needs to be conducted which specifically addresses each of these three components and their use in software training

Implications for the Field

Having its roots firmly grounded in the fields of adult education and training; Human Resource Development (HRD) is perhaps the ideal place to present research on adult computer software training. Yet little research has been published specifically in this field. In the last 10 years, Human Resource Development Quarterly had only 4 articles that dealt with software training. Three of those articles had no relevance to this topic and the fourth article has been cited in this paper. Additionally, with the integration of computers into nearly every setting and the ever changing and updating nature of software; HRD scholars and practitioners alike would be well suited to learn the essentials of adult computer software training. Thus this study could provide a beginning framework for further exploration of the adult computer software training process.

References

Carroll, J. (1990). *The nurnburg funnel: Designing minimalist instruction for practical computer skill*. Cambridge, MA: The MIT Press.

Coffey, A., & Atkinson, P. (1996). *Concepts and coding*. Thousand Oaks, CA: Wadsworth.

Egan, T. M. (2002). Grounded theory research and theory building. *Advances in Developing Human Resources*, *4*(3), 277-295.

Gupta, A., & Ndahi, H. (2002). Meeting the digital literacy needs of growing workforce. *Reading Matrix: An International Online Journal, 2*(1), Retrieved on 9/17/2006 from http://www.readingmatrix.com

Hansman, C. (2001). Context-based adult learning. In S. Merriam (Ed.), *The new update on adult learning theory* (Vol. 89, pp. 43-51). San Francisco: Jossey-Bass.

Harp, C. G., Taylor, S. C., & Satzinger, J. W. (1998). Computer training and individual differences: When method matters. *Human Resource Development Quarterly, 9*(3), 271-283.

Hughes, M. A. (1998). Active learning for software products. *Technical Communication: Journal of the Society for Technical Communication, 45*(3), 343-352.

Knowles, M. (1968). Andragogy, not pedagogy. *Adult Leadership, 16*(10), 350-352, 386.

Lambrecht, J. J. (1999). Teaching technology-related skills. *Journal of Education for Business, 74*(3), 144-151.

Lambrecht, J. J. (2000). Developing end-user technology skills. *Information Technology, Learning, and Performance Journal, 18*(1), 7-19.

Mathison, S. (1988). Why triangulate? *Educational Researcher, 17*(2), 13-17.

Merriam, S. (1998). *Qualitative research and case study applications in education*. San Francisco: Jossey-Bass.

Merriam, S., & Caffarella, R. (1999). *Learning in adulthood* (2nd ed.). San Francisco: Jossey-Bass.

Patton, M. (1988). *Qualitative evaluation and research methods (2nd ed.)*. Thousand Oaks, CA: Sage.

Peshkin, A. (1988). In search of subjectivity-one's own. *Educational Researcher, 17*(7), 17-21.

Tomorrow's jobs. (2003). *Occupational Outlook Handbook.* Retrieved on 9/17/2006 from http//www.bls.gov/oco/oco2003.htm

Wilson, A. (1993). The promise of situated cognition. In S. Merriam (Ed.), *An update on adult learning theory* (Vol. 57, pp. 71-79). San Francisco: Jossey-Bass.

Wolcott, H. (1994). *Transforming qualitative data*. Thousand Oaks, CA: Sage.