U.S. Army Research Institute
for the Behavioral and Social Sciences

Research Report 1906

Determining a Critical-Skill Hierarchy for Command Post of the Future (CPOF)

Richard Catrambone
Georgia Institute of Technology

Richard L. Wampler
Northrop Grumman Corporation

Martin L. Bink
U.S. Army Research Institute

July 2009

U.S. Army Research Institute
for the Behavioral and Social Sciences

A Directorate of the Department of the Army
Deputy Chief of Staff, G1

Authorized and approved for distribution:

Barbara A Black

BARBARA A. BLACK, Ph.D.
Research Program Manager
Training and Leader Development Division

Michelle Sams

MICHELLE SAMS, Ph.D.
Director

Research accomplished under contract
for the Department of the Army

Northrop Grumman Corporation

Technical review by

Gregory A. Ruark, U.S. Army Research Institute
Jon Fallesen, U.S. Army Research Institute

NOTICES

REPORT DOCUMENTATION PAGE

1. REPORT DATE (dd-mm-yy) July 2009	2. REPORT TYPE Final	3. DATES COVERED (from. . . to) June 2008 – May 2009

4. TITLE AND SUBTITLE Determining a Critical-Skill Hierarchy for Command Post of the Future (CPOF)	5a. CONTRACT OR GRANT NUMBER W74V8H-04-D-0045 DO#0030
	5b. PROGRAM ELEMENT NUMBER 633007
6. AUTHOR(S) Richard Catrambone (Georgia Institute of Technology), Richard L. Wampler (Northrop Grumman Corporation), Martin L. Bink (U.S. Army Research Institute)	5c. PROJECT NUMBER A792
	5d. TASK NUMBER 359
	5e. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Northrop Grumman Corporation, 3565 Macon Road, Columbus, GA 31907; US Army Research Institute for the Behavioral and Social Sciences, ARI-Ft Benning Research Unit, P. O. Box 52086, Fort Benning, GA 31995-2086	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) U. S. Army Research Institute for the Behavioral & Social Sciences, ATTN: DAPE-ARI-IJ, 2511 Jefferson Davis Highway, Arlington, VA 22202-3926	10. MONITOR ACRONYM ARI
	11. MONITOR REPORT NUMBER Research Report 1906

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

Contracting Officer's Representative and Subject Matter POC: Martin L. Bink

14. ABSTRACT *(Maximum 200 words)*: The Command Post of the Future (CPOF) is a dynamic visualization tool that supports collaborative decision-making in tactical units. The system uses a customizable workspace based on the user's needs rather than a static format. While such an approach to digital-systems design offers flexibility and generality of use, it might also increase the complexity of learning to use the interface. As a precursor to examining alternative training approaches for CPOF, this report documents an analysis of and hierarchical structure for underlying CPOF skills. A knowledge extraction process was conducted with CPOF domain experts (DEs) to uncover the knowledge needed to use CPOF. The DEs performed a series of tasks based on the practical exercises developed for training Soldiers. A Critical Skills Document was iteratively updated and reorganized in order to identify the major components of the system and the procedures for accomplishing various tasks. The Critical Skills Document represents CPOF skills in a way to show their generality and applicability. Instructional designers can use it to determine what to train as well as a guide for developing learning assessments. The findings provide a foundation for comparing training approaches for CPOF and similar digital systems.

15. SUBJECT TERMS
CPOF Command Post of the Future Task Analysis Skill Hierarchy Knowledge Elicitation Command and Control Training Analysis Knowledge Extraction Critical Skills

SECURITY CLASSIFICATION OF			19. LIMITATION OF ABSTRACT	20. NUMBER OF PAGES	21. RESPONSIBLE PERSON
16. REPORT Unclassified	17. ABSTRACT Unclassified	18. THIS PAGE Unclassified	Unlimited	46	Ellen Kinzer Publications Technician Special 703-602-8049

Research Report 1906

Determining a Critical-Skill Hierarchy for Command Post of the Future (CPOF)

Richard Catrambone
Georgia Institute of Technology

Richard L. Wampler
Northrop Grumman Corporation

Martin L. Bink
U.S. Army Research Institute

ARI – Fort Benning Research Unit
Scott E. Graham, Chief

U.S. Army Research Institute for the Behavioral and Social Sciences
2511 Jefferson Davis Highway, Arlington, Virginia 22202-3926

July 2009

Army Project Number	**Personnel Performance**
633007A792	**and Training**

ACKNOWLEDGEMENT

Many personnel contributed to this research effort and without their contributions this critical skills hierarchy would not exist. Most importantly, the expertise provided by the various CPOF trainers at Forts Hood (Mitchell Carlisle and Guillermo Quintero) and Benning (Barry Wilson) served as the basis for this analysis. Their insightful comments and precious time were immeasurable. At various times throughout the project other Northrop Grumman personnel made significant contributions to assist with the knowledge elicitation sessions and to conduct follow-up information gathering with CPOF trainers, as well as to offer input into the task analysis process and task hierarchy. David James, Norm Blankenbeckler, and Michael Dlubac each provided key support to assist with project execution. We are indebted to Drs Jean Dyer, Jennifer Murphy, and Marisa Miller for valuable comments on previous drafts of this Report.

DETERMINING A CRITICAL-SKILL HIERARCHY FOR COMMAND POST OF THE FUTURE (CPOF)

EXECUTIVE SUMMARY

Research Requirement:

Many of the Army's digital systems depend more on learning a skill set as opposed to learning task execution. This is particularly the case with Command Post of the Future (CPOF). CPOF is a dynamic visualization tool that supports collaborative decision-making in tactical units. CPOF uses a customizable workspace that is based on the user's needs rather than a static data format. While such an approach to digital-systems design offers more flexibility and generality of use, the flexibility of CPOF might also increase the complexity of learning to use the interface. As a precursor to examining alternative training approaches for CPOF, a full understanding of underlying CPOF skills and the hierarchical structure of those skills is required. The goal for this research effort was to analyze and document critical CPOF skills by means of a knowledge elicitation methodology. The findings provide a foundation for future research comparing training approaches for CPOF and similar digital systems.

Procedure:

Two separate knowledge extraction sessions were conducted, one each at Fort Hood and Fort Benning. In each case the knowledge extraction expert (KEE) worked with a domain expert (DE) to uncover the knowledge needed to use CPOF. The DE performed a series of tasks based on the practical exercises developed for training Soldiers. The KEE took detailed notes and continually required the DE to explain why he was doing each step. The KEE tested the accuracy of the notes (called the Critical Skills Document) by doing tasks provided by the DE. The Critical Skills Document was iteratively updated and reorganized by the KEE and other members of the research team in order to identify the major components of the system and the procedures for accomplishing various goals.

Findings:

The resulting Critical Skills Document covers the major capabilities of CPOF and organizes them in a way that makes the relations, including hierarchical ones, among those capabilities clear. It identifies the various procedures and sub-procedures needed to use CPOF and represents them in a way to show their generality and applicability.

Utilization and Dissemination of Findings:

The Critical Skills Document can be used by instructional designers to determine what to teach learners as well as a guide for developing assessments of learning. A key value of this document is that it makes explicit the information that must be conveyed to learners; this is often information that has become internalized by experts and as a result is not explained clearly--or at all--to novices.

DETERMINING A CRITICAL-SKILL HIERARCHY FOR COMMAND POST OF THE FUTURE (CPOF)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

DETERMINING A CRITICAL-SKILL HIERARCHY FOR COMMAND POST OF THE FUTURE (CPOF)

Introduction

Many of the Army's digital system training programs depend more on learning a skill set as opposed to learning task execution. Learning a skill set is particularly the case with the Command Post of the Future (CPOF). CPOF is touted as a dynamic visualization tool that supports decision-making in tactical units in a collaborative environment. Commanders and staff officers at brigade and battalion level can use CPOF to share information, such as operations overlays while they are being developed, which assists them in planning tactical operations. Staff personnel can also use CPOF to monitor battlefield operations and provide update briefings to leaders. To accomplish these functions, the CPOF interface uses a customizable workspace that is based on the user's needs rather than a static data format. Successful application of CPOF requires the user to decide which functions will best address a problem or need. Thus, digital skills might be combined in a non-linear fashion rather than in a step-by-step sequence typical of many Army tasks.

As a brief overview, CPOF is intended to work with three monitors arranged as shown in Figure 1. The cursor moves among all three monitors as though across one large, continuous workspace. A user can configure and arrange the entire workspace, as desired, to facilitate personal preferences and task execution. CPOF displays battle information on maps, in tables and in schedules to allow decision-makers to "visualize" relevant information in the most useful manner. Users "construct" the appropriate display formats from a set of digital tools provided by CPOF. Users also may access and share information through a common information portal and may communicate with other users through network capabilities and voice-over-internet protocol.

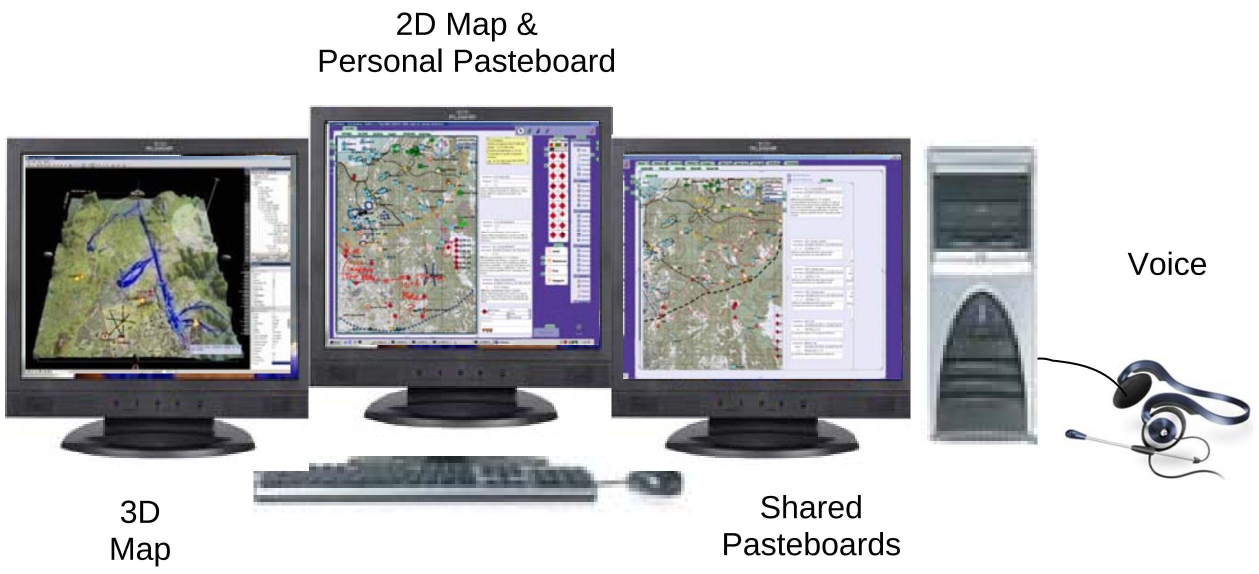

Figure 1. CPOF 3-Monitor Workspace.

The purpose of this research effort was to produce a conceptual framework for CPOF critical skills that can lead to the development of training approaches for CPOF. The conceptual framework was based on the identification of critical CPOF skills and of the hierarchical structure among skills. As such, the results of this research were intended to be used by CPOF training developers and CPOF trainers as a resource to plan programs of instruction and to assist in the modification of materials. This report also contains the documentation of a knowledge extraction process (i.e., task analysis) that can be used more generally by training personnel to develop new training materials.

Background

Because the CPOF interface is mostly non-linear (i.e., interaction with the system is not based on prescribed sequences of steps and data), there is less internal cuing in the interface. Proficiency with non-linear interfaces requires a higher-level of understanding of task goals and interface capabilities (Farrell & Moore, 2000). Traditional instructional approaches (e.g., lecture and practical exercises) might not help a learner to develop such higher-level understanding as efficiently as other learning approaches.

While there is little empirical research on the training requirements of non-linear interfaces, the research on learning from hypertext is an analogy for training non-linear digital-system interfaces. Two particular problems noted in research on learning from hypertext are (a) that learners (i.e., users) can become disoriented as they click link after link into the text and get further away from the starting point (e.g., Chen, 2002; Ellis & Kurniawan, 2000) and (b) that learning involves an independent and active learning process (Chen, 2002; Ford & Ford, 1993). Both of these problems seem to be related to the learner's level of conceptual knowledge of the task. That is, given that there is flexibility in the manner in which tasks are completed, people who have hierarchical task knowledge are better able to monitor task progress and to select alternatives that will lead to efficient task completion (Chen, 2002; Dunlosky & Lipko, 2007).

Thus, the difficulty in learning non-linear digital interfaces such as CPOF appears to be based on the fact that novices do not have, and the interfaces do not support, the organization of knowledge necessary to successfully interact with the system. One possible solution to the potential problems of learning non-linear digital system interfaces is to base training on the development of hierarchical conceptual knowledge instead of on the memorization of steps (cf. Newell & Simon, 1972). The difficulty with this approach is that the development and use of hierarchical knowledge is associated with expert performance (Larkin, McDermott, Simon, & Simon, 1980) rather than as a method of training novices. However, some studies have demonstrated that novices benefit from learning hierarchical knowledge as compared to learning step-by-step procedures (e.g., Catrambone, 1998; Dufresne, Gerace, Hardiman, & Mestre, 1992).

Hierarchical knowledge of digital systems includes the steps necessary to support the execution of a given procedure, the structure of the procedure, the underlying purpose of the procedure in executing a task, and some understanding of how the system functions as a whole. In general, developing hierarchical knowledge involves learning meaningful components of the overall concepts and then structuring those components based on the requirements of task goals (cf. Catrambone, 1998). In the case of learning non-linear digital systems such as CPOF, it is assumed that developing hierarchical knowledge is based upon learning the skills that are most critical to the intended functions of the system, applying those critical skills in task execution, and structuring the critical skills based on the application of those skills *across* tasks. In this case, “skills” refer to manipulations of the system interface that support completion of multiple tasks. It is this type of skill that is the focus of this analysis.

Most Army digital systems are similar to a personal computer. The operator learns appropriate key strokes, becomes familiar with drop-down menus, and memorizes selected processes that are frequently used. This technique does not work for CPOF. In order to avail oneself of the CPOF capabilities, the system user must know the appropriate key strokes and menus but, more importantly, must also learn the total system capability and understand how the CPOF system can be applied to meet the user needs. The operator or user must know and understand how to apply CPOF to aid in battlefield visualization, decision-making, and problem solving. That is, the user must not only know *how* to do things, but also know *when* to do them (i.e., decision rules).

The ultimate goal for Army digital system trainers is to ensure that Soldiers learn how to operate and employ the family of digital systems to help them accomplish missions on the battlefield. Trainers want Soldiers to be able to solve novel problems, that is, problems that are not just like the examples presented in lecture or in the reference book. Soldiers need to be able to apply the procedures described to new situations. In short, trainers should strive to ensure that users can employ the digital system capabilities to improve the acquisition and transfer of problem solving and procedural knowledge.

Research directed at transfer of problem solving and procedural knowledge has tended to focus on ways of improving training exercises, the use of technology to aid learning, and individual differences in learning styles. These studies have produced many conclusions and claims about materials and learners. These claims and conclusions often contradict each other. For instance, some studies have shown that weaker learners benefit more from training materials that make use of multiple modalities and media while other studies have shown that it is primarily stronger learners who can best benefit from such enhanced instructional information (Kalyuga, Chandler, & Sweller, 1999; Mayer, 2005).

Why do such contradictions exist? One key reason might be because the quality of the training and testing materials used in such studies has been variable. That is, the materials have often failed to contain important information that learners needed to know in order to solve new problems or carry out a procedure. As a result, the instructional manipulations examined in these studies and any conclusions drawn from the results are suspect. In earlier research, Catrambone (1995, 1996, 1998) reworked training examples developed by other researchers for their problem solving studies (e.g., Reed, Dempster, & Ettinger, 1985; Ross, 1989; Ross & Kennedy, 1990) to make sure they included key information identified through a task analysis. Catrambone found that learners' subsequent transfer to novel problems was superior to that found in the prior studies and that in some cases the training manipulations borrowed from other studies no longer had an impact--or had a different impact--on learning.

Technical Objectives

The primary objective in this work was to create a conceptual framework for CPOF critical skills that can lead to the development of effective training approaches for CPOF. The essence of the conceptual framework was the identification of critical CPOF skills and of a hierarchical structure among tasks. The identification of the CPOF skills was accomplished by using the task analysis approach or knowledge elicitation method described in the following section. By identifying the appropriate structure for critical skills, trainers will be able to develop and present training programs that will lead to enhanced CPOF system performance and readiness in operational units.

The following technical objectives guided the research:

- Identify the critical skills required to operate CPOF.
- Analyze critical skills to determine the hierarchical structure among CPOF tasks.

By creating a critical skills structure, this research paves the way for follow-on efforts to design and develop training approaches that optimize learning for CPOF users.

A key feature of the task analysis approach used for the current project is that it does not rely on a complex model of cognition. That is, the many assumptions about cognitive operations--such as claims about memory, chunking, and schema formation--do not seem to account for much of the performance related to learning from examples or other realistic training materials (Catrambone, 1994, 1995, 1996, 1998). Models of human cognition might provide guidance on *presentation* issues, but they do not provide obvious constraints on *what* information to present. For example, claims about memory organization and procedural learning are unlikely to tell us anything about how to identify the problem solving knowledge needed to solve mechanics problems in physics. Models can provide tools for representing information to-be-learned and this notional value might be their most useful contribution to the design of teaching and training materials. For instance, one approach to task analysis is to create a set of production rules that solve problems or carry out procedures that one wants learners to be able to solve or learn (e.g., Anderson, Boyle, Farrell, & Reiser, 1987; Kieras & Bovair, 1986; Zhu & Simon, 1987). However, it does not seem necessary to make a commitment to a production rule embodying a particular learning theory such as ACT (Anderson, 1983) or Soar (Laird, Newell, & Rosenbloom, 1987) in order to derive the elements that need to be learned. Still, a fundamental feature of most production rule systems--the goal structure--does provide a useful way to represent the knowledge needed to solve problems in a domain. Subgoals show the breakdown of a problem-solving procedure into sub-problems (Anzai & Simon, 1979). Thus, the current task analysis approach differs from other cognitive task analyses in its relative simplicity and its emphasis on subgoals.

Description and Rationale for Task Analysis Approach

An expert in a domain--regardless of whether the domain is repairing carburetors, pitching baseballs, or designing houses--is, of course, very good at the tasks in that domain. However, a cost of expertise is that experts are often unable to describe *how* and *why* they do the various steps that make up a task. This cost is due to the fact that, for an expert, many parts of tasks have become automated, many steps seem obvious, and many cues for guiding the choice of steps (and even the choice of task) can not be articulated easily. The aim of the task analysis approach developed by Catrambone (e.g., Catrambone, 1998; Gerjets, Scheiter, & Catrambone, 2004) is to uncover/rediscover the knowledge that an expert uses. The value of recovering such knowledge is that it can form the basis for developing, and be integrated into, instructional materials for new learners as well as learners at various levels of expertise.

Problems within a domain typically share the same subgoals; a *subgoal* represents the purpose of a set of steps. The steps for achieving those subgoals will differ across problems. For instance, consider algebra word problems dealing with work. One problem might be:

> Joe can paint a fence in three hours; Mary can paint it in two hours. How long will it take them to paint the fence if they work together?

Another problem might be:

> Sue can wash a car in 1 hour. Steve can wash a car twice as fast as Sue. How long will it take them to wash a car if they work together, but Sue starts 30 minutes before Steve?

These problems can not be solved using the exact same set of steps, but they share the same subgoal structure which includes representing each worker's work rate and representing how much time each worker works.

Clearly the scope of the domain plays a role in how useful or meaningful the subgoals will be. In general, the term "domain" refers to the information comprising a coherent instructional unit such as the information contained in a chapter (or perhaps half-chapter) of an introductory physics or chemistry textbook. For instance, in one set of probability training experiments Catrambone conducted (e.g., Catrambone, 1994) the domain was permutation and combination problems; interestingly, one might argue permutation and combination problems are separate units, but the subgoal analysis indicated they share the same set of subgoals.

A variety of task analysis techniques exist (for a review see Schraagen, Chipman, & Shalin, 2000). The merits and pitfalls of these techniques have been considered by multiple critics. Pitfalls include unnecessary complexity and overly narrow application (e.g., a given task analysis technique might not be useful for identifying procedures) (Schraagen et al., 2000). Catrambone (1998) developed a task analysis approach that is relatively conceptually simple--although requiring a good deal of detailed consideration of the materials and an iterative approach with those materials--and is well-suited to identifying the procedural knowledge needed to carry out tasks and/or solve problems in a particular domain. This approach has been applied successfully in a variety of domains--albeit most with an academic bent--in order to develop instructional materials in experimental settings (Catrambone 1994, 1995, 1996, 1998). The domains have included probability, physics, ballet, computer algorithms, and chemistry.

Conducting a task analysis is not the same thing as developing learning objectives. That is, a teacher or trainer might have a learning objective that students be able to solve algebra word problems dealing with work (such as those previously provided). However, such learning objectives say little about the procedural content the student needs to know in order to solve problems (e.g., represent each worker's work rate; represent how much time each worker works; multiply each worker's rate by the worker's time; etc).

Different people will possibly disagree on what constitutes the "right" things that a learner needs to know and this is why it is important to work with multiple domain experts (DEs) to extract the knowledge used to carry out tasks. Typically there is an overlap in the extracted knowledge which can become the basis for the instructional material to be developed. Even in cases of rough agreement, there will be differences in what level of detail to include in the task analysis. For instance, in the algebra word problem described earlier, the task analysis states that one of the things the learner needs to know how to do is to represent each worker's rate. This piece of knowledge though could be broken down into smaller pieces concerning how the fraction might be constructed or how the rate should be handled if the rate is unknown. Decisions about how low level to get in the task analysis itself will be a function of a variety of factors including the assumed background of the learners, the time allotted to cover the material, the goal of the instructor, etc.

It is important in the task analysis approach Catrambone (1998) has developed that a domain expert (e.g., a CPOF trainer) and a domain novice, but knowledge extraction expert (KEE) work together. The DE identifies a set of typical problems, tasks, and/or scenarios that novice learners are to be able to solve or carry out. Once the list of training problems to solve has been identified, execution of the knowledge extraction (KE) session then typically proceeds as follows:

- The **DE solves the problems (or carries out the tasks) with the KEE observing**. The KEE's job, as a domain novice, is to require the DE to explain and defend the steps and decisions the DE makes as he solves the problems. The aim here is not just to identify the steps, but to **create detailed notes to explain *why* each step is being carried out** (and sometimes why a different step was not chosen) and, often, the subgoals being achieved by particular groups of steps.

- After developing the notes, the KEE attempts to **solve additional problems or carry out additional tasks** identified by the DE. This allows the KEE to verify the accuracy and completeness of the notes, often leading to elaboration of the notes. Inevitably the KEE will reach an impasse on a particular problem, that is, when the KEE's state of knowledge (represented by the notes) does not allow the KEE to determine what to do next. At that point the KEE consults with the DE. Typically such consultations lead the DE to recall some piece of information that had not been explicitly identified previously. The DE then supplies that bit of information. Sometimes a misunderstanding by the KEE is uncovered, which can be corrected through consultation with the DE.

- The KEE **revises his notes based on the feedback and continues to solve more problems** until the KEE can solve all problems/tasks in that domain that were supplied by the DE. Not surprisingly, this is an iterative process. During this process the procedural information becomes less tied to details of the specific problems solved by the DE and KEE.

- Subsequently, the KEE **divides the notes into categories** such as subgoals, steps, facts, rules, definitions, implications, and conventions. Different domains might lend themselves to different characterizations of problem solving knowledge types. While there is evidence that subgoals have some psychological validity for predicting problem solving transfer (Catrambone, 1996, 1998), the other categories are more of an organizational tool.

After carrying out the above iterative process, the KEE can conclude that he has identified all of the needed information to solve problems or carry out tasks in the domain. While it does not constitute a formal proof, this iterative approach has worked well. It is important to recognize that the information is not derived--and probably can not be derived--from a formal analysis of the domain. A formal analysis would not uncover the problem solving procedures.

Method

The KE approach previously described served as the basis for this analysis. The goal of the analysis was to identify critical CPOF skills and the relations among the skills in order to develop a Critical Skills Document that could guide the subsequent development of CPOF training materials and assessment materials. In order to satisfy this goal, the analysis was conducted in three phases. The first phase used the knowledge extraction procedure to identify the critical CPOF skills. The second phase refined and reviewed the critical skills generated in the first phase in order to construct the Critical Skills Document. In the third phase, the critical skills were formalized into the organized structure of the final Critical Skills Document.

Participants

In each phase, researchers collaborated with DEs to build the knowledge base of critical skills. All researchers were CPOF novices with limited knowledge of the digital system. The researchers included a KEE who used the KE process in the past and was expert in the process. The researchers also included experts in the Army's military decision-making process. The team was structured so that the KEE could identify the CPOF critical skills and the Army experts could ensure the identified critical skills would lead to the desired applicability to operational employment of CPOF in tactical units.

The DEs were CPOF trainers from the BCTC at Fort Hood, TX and from the digital training facility at Fort Benning, GA. Each DE was primarily responsible for CPOF classroom training, but each also had experience with field training and with the deployment of the system. Each DE had over one year's experience training CPOF as well as prior military experience.

Procedure

Identification of critical skills. The initial step was to acquire some background knowledge on this digital system. Researchers reviewed the most current (at the time) CPOF User Guide. They also obtained and reviewed various Practical Exercise documents being used by expert trainers. The intent was to gain some initial exposure to the CPOF system without contaminating the KEE with too much knowledge about the system. That is, the KEE needed to remain a novice but also needed to have a broad overview of CPOF.

As per Catrambone's (1998) task analysis approach, the initial step in the process was to have the DE identify a set of typical problems, tasks, and/or scenarios that learners are to be able to solve or carry out. Existing "Practical Exercises" used in training new users were chosen given that they were developed by CPOF trainers (i.e. DEs) to provide new users an exposure to many aspects of the CPOF system.

Two KE sessions were conducted over two days at Fort Hood. The DE initially completed some training exercises from the Practical Exercise documents that were in use for training new CPOF users. Version 3.0.2P2 of the CPOF system software was used. While the DE did the initial exercises, the KEE took detailed notes of what steps the DE was performing and, importantly, the reasons and justifications for the steps. The KEE would frequently ask the DE why he was doing certain steps and what the conditions were for doing those steps. These notes served as the foundation for the Critical Skills Document. At the end of each day the KEE revised notes for clarity, corrected inconsistencies, and identified any missing knowledge. The KEE also modified the notes to make them more generally applicable rather than tied to the specific exercises being demonstrated. During this process the KEE would attempt various exercises using the evolving Critical Skills Document as a guide. The DE was available to answer questions. The KEE continued to revise the Critical Skills Document in an iterative fashion based on these interactions.

While not a formal step in the KE process, following the KE sessions, the KEE had an opportunity to observe several hours of CPOF classroom training. This allowed the KEE to learn about the approaches used in the class as well as to observe the performance of the learners.

Analysis and refinement of Critical Skills Document. After the Fort Hood KE sessions, the Critical Skills Document was revised with several aims in mind. First, and foremost, the most critical CPOF skills needed to be identified. Second, factual errors were corrected, and some incomplete sections in the Critical Skills Document were expanded. Third, the goals and procedures delineated in the document needed to be portrayed in a more general nature. Fourth, goals and procedures were analyzed in order to capture the main and subordinate goals in addition to the task procedures in order to highlight the relations among the various goals, subgoals, and procedures. Such an analysis has important implications for training CPOF skills because it can help instructors present material in a way that emphasizes the organization of CPOF for the learner. The premise was that the better the organization of the goals, the more efficient the learning, the better the knowledge retention, and the better the ability of the learner to flexibly apply and generalize the information to new situations.

A second KE session was conducted with a CPOF trainer at Fort Benning. Due to the non-linear structure of the CPOF system, and the limited time trainers have to present the material, it was necessary to ensure the completeness of the critical skills contained in the Critical Skills Document as well as to see if the goal structure might change due to another expert's perspective. During this KE session, the KEE completed more of the Practical Exercises used in training new users as a way of checking the accuracy and organization of the Critical

Skills Document. The DE was available to answer questions and to provide additional detail concerning steps for completing procedures. Following this KE session, some details of the Critical Skills Document were periodically confirmed with the DEs to ensure accuracy. This process led to further refinements of the Critical Skills Document. It is worth noting that the CPOF software was upgraded to version 4.1 during this period, but none of the DEs indicated that the change in software caused any changes to the goals or procedures in the Critical Skills Document.

Identification of a hierarchical structure of CPOF critical skills. As refinements continued to the Critical Skills Document, a schematic architecture to represent the major functional capabilities of CPOF and their relations was developed. This schematic had two roles. First, it provided a visual portrayal of the organization of the existing Critical Skills Document. Second, it became a tool that allowed us to review relations among the CPOF procedures. The schematic provided a means to quickly assess the relations and then reorganize the procedures as deemed appropriate. The schematic also provided an additional check to ensure the necessary critical skills along with their required procedures were identified.

The development of the schematic architecture became an iterative process with the research team occasionally consulting a CPOF User's Guide and confirming emerging changes with the DEs. Because the Critical Skills Document contained the goals, subgoals, and procedures for the CPOF critical skills, the Critical Skills Document became the basis for the hierarchical structure. Likewise, as the schematic evolved, it was used to revise the organization of the Critical Skills Document. Throughout development of the Critical Skills Document, efforts were made to ensure entries were generally applicable to multiple tasks, rather than tied to the specific exercises conducted during the KE sessions. These efforts allowed the resultant hierarchy to be applicable to the entire scope of CPOF critical skills.

The development of the skill hierarchy was focused on the critical skills required for users to operate the CPOF system in an operational environment. This meant that the users could employ the CPOF functional capabilities to accomplish dynamic visualization while conducting decision-making in a collaborative environment. The intent was to create a hierarchy that represented the critical skills of the CPOF system, without regard for the military decision-making expertise of the user being trained.

Product of the Research

The Critical Skills Document is presented in the Appendix. The Critical Skills Document identifies the knowledge needed for the major procedures in CPOF. The information has been organized hierarchically when possible and has also been divided into categories representing major divisions of types of knowledge. The aim here is to provide the knowledge for a designer to create training materials and exercises that will capture the major CPOF features.

Consider the CPOF procedure called an "Effort." In CPOF, an Effort is a generic container, much like a work folder in computer systems. The "Effort" can be used to gather all the work products a user might need while working on a project or a plan. Efforts do not have a particular geographic location on a map or time of occurrence; they do not have a limit for the types of products that they can contain. The properties of the Effort determine how the elements are displayed in CPOF.

Table 1 is an excerpt from the Critical Skills Document that pertains to Efforts. The excerpt describes the procedure for creating an Effort and includes information and commentary to make explicit the steps as well as things for the user to notice when doing the task. The procedure references the “Frame Dispenser” which is a part of CPOF that is described and defined earlier in the Critical Skills Document. This last observation highlights two important aspects of the Critical Skills Document: hierarchicality and procedural overlap. Various procedures appear as sub-procedures in the service of a higher procedure. So, in this example, the procedure of accessing the Frame Dispenser and getting items from it is described earlier in the Critical Skills Document and thus is available to be referenced in subsequent procedures--like creating an Effort--when needed.

Table 1
Excerpt from Critical Skills Document for “Effort”

Efforts
• An Effort can effectively be considered a Work Folder
• Each Map has an Effort List that is always present and shows all the Efforts associated with that Map
• Note that items from geographically different areas can be put into the same Effort
– To create an Effort
- Get *Effort palette* from Frame Dispenser
- In Effort palette, drag out *Effort*
- Left click on title of Effort box and type in the desired title
- Notice box “contain elements here”
- Left click and then grab each desired item from Map and drag into the “contain elements” box
- Alternatively can left click and drag across all the items and they will get highlighted and the user can drag them all to Effort window
- Dragging items to Effort window will preserve them for that Effort
- Drag the named Effort and drop it into Effort List of the Map

The Critical Skills Document attempts to take advantage of overlap among procedures. That is, many procedures are conceptually identical or similar. Thus, it is to a learner's advantage to learn these similarities so that his or her knowledge will be organized more generally rather than as a set of disconnected facts and procedures. This overlap will also reduce the amount of information the learner needs to acquire; instead, the learner can use his understanding of the general procedures to recall, reconstruct, or generate procedures when needed.

The similarity among procedures was captured in the Critical Skills Document by writing the steps for various procedures to emphasize their overlap. Consider a Soldier who wants to create a Unit on a map. The procedure for doing this is shown in Table 2 taken from the Critical Skills Document.

Table 2
Procedure for Creating a Unit in CPOF and Placing on a Map

– Create a Unit
– Get Unit/Event Palette from Frame Dispenser
– Click on Unit tab (if not already the selected tab)
– Drag any Unit to desktop
– Use BACKSPACE key to remove "untitled" and give it name (e.g., 2BCT)
– Type grid coordinates into grid coordinate field
– Select type of Unit (e.g., infantry)
– Use drop down boxes to select features (e.g., blue, friendly, brigade)
– Drag the window to the 2D Map anywhere
- It might disappear if grid location for the Unit is not on current Map, but if the user goes to Map with that grid location on it, the Unit will be displayed there
• Can make several of these and Clone each on to Map

Now, consider a Soldier who wants to place an "Event" on the map. Events are typically used to define significant activities in CPOF, but they can serve other purposes as well. An Event has a name, grid location, a "type" (such as improvised explosive, mortar attack, etc.), date and time, and comments or description. Events can also have associations; for example, a vehicle-based improvised explosive could be associated with the route of the vehicle prior to detonation. The procedure for creating an Event is shown in Table 3.

Table 3
Procedure for Creating an Event in CPOF

– Create an Event
– Get Unit/Event Palette from Frame Dispenser
– Click on Event tab (if not already the selected tab)
– Drag any Event to desktop
– In resulting box, fill in information (title which is usually date and time)
– Add info in drop down boxes
- e.g. In "Type" the user might select "small arms fire (SAF)"
– Type grid in the GridCoords box
– In the Comments field, type summary of Event itself (in user's own words)
– Drag and drop Event box to the 2D Map anywhere
- It might disappear if grid location for the Event is not on current Map, but if the user goes to Map with that grid location on it, the Event will be displayed there

What is seen is that these procedures have a great deal in common and in some cases the steps are identical. What differs are some details about the information that users would enter in the drop down boxes that appear at various steps. By emphasizing common steps, the learner can learn more quickly. These two procedures, creating a unit and creating an event, highlight another important benefit of the task analysis approach described in this document. Sometimes DEs will mention certain strategies or details in an almost offhand way that a KEE applying this task analysis procedure will recognize as important for a novice. In this case, the issue was the placement of the Unit or Event on a map. In working with the DEs it became clear that it really did not matter what map was present on the screen when a Unit or Event (and by extension, other graphical products) was created because in the process of creating these entities, one specifies the grid location. After the entity is created, one can go to the appropriate map, if desired, to verify or adjust the location. This issue might seem like a small point, but it became clear that this was how an expert worked with the system. It was also equally clear that this approach was not something an expert might explicitly mention or justify unless pushed by a KEE.

The procedural similarities resulted from many re-workings of the Critical Skills Document. The Critical Skills Document started as step-by-step descriptions of how to do the various practical exercises supplied by the CPOF trainer; the KEE took notes that pertained to each exercise with some on-the-fly abstractions as seemed appropriate. However, the Critical Skills Document was then revised in order to extract the overlapping steps between or among similar procedures.

For example, consider the section of the Critical Skills Document on graphics which is in the “Construct” part of the Critical Skills Document. Initially there were detailed steps for a variety of graphics exercises that were in the Practical Exercises documents. An analysis of the various exercises led us to identify three types of graphics: point, linear, and area graphics. These three types are listed under the general graphics heading. The Critical Skills Document first lists some basic commonalities to constructing graphics--for example, getting the Toolbar and the Graphics Palette from the Frame Dispenser--and then goes into each type. For each type of graphic, the steps are given for how to create a particular graphic, for example, how to create a Main Supply Route (a type of linear graphic). The use of specific examples for each graphic type accomplishes three objectives. First, it provides a concrete example that shows step-by-step what to do. Second, by describing the basic commonalities among the graphics procedures, this information can be leveraged to guide the generalization of the individual examples to other graphics within a particular graphic type. So, for instance, by specifying the steps for drawing a Main Supply Route and for drawing a Boundary, and given the general procedure for producing graphics, one should have enough information to infer how to draw other linear graphics. Third, if certain procedures have special features or inconsistencies with other procedures in the category, these can be illustrated in the example. For instance, the procedure for drawing an air path--a type of “area graphic”--requires that one use the 3D screen while most (if not all) other area graphics can be drawn on the 2D or 3D screen. Thus, the air path example was important to include in the Critical Skills Document so that this feature was highlighted.

One challenge in developing the final Critical Skills Document was determining how to represent aspects of the system when a particular goal or procedure needed to reference some part of the system that had not yet been described. For instance, the process of Cloning--a key functional aspect of CPOF whereby the content of a frame (such as a table, chart, or map) is copied--is mentioned early in the Critical Skills Document and presumably should be mentioned early in CPOF training given its centrality to CPOF. However, given the complexity of Cloning, it would not make sense to describe the process in detail early in learning. Such information would presumably overwhelm a learner because he or she would not yet have the necessary mental structure from which to “hang” such information. Therefore, the Critical Skills Document occasionally references a concept or procedure without explaining it in detail at the initial mention. Such mentions are footnoted in the Critical Skills Document with a reference to where the details of the concept or procedure are described.

The procedural similarities are also captured in the overall hierarchical organization of the Critical Skills Document. The notion of a hierarchy was used in the Critical Skills Document in an attempt to capture the overall structure of the CPOF system. The skill hierarchy was intended to accomplish two purposes. First, the hierarchy needed to reflect an organization that was *functionally* sensible. Second, the skill hierarchy would need to be structured so it could guide the development of training materials that would maximize learning efficiency, retention, and problem solving flexibility. Multiple hierarchies that divided the CPOF capabilities and skills to emphasize different functional aspects of the system were considered. For instance, an organization based on military goals or an organization that focused on software capabilities regardless of the domain of application could be used. However, the most practical organization was one based on the construction, display, and sharing of tactical "products" within CPOF. The thought process is that the user would first need to create or construct some tactical product (e.g., a map overlay). The completed product could then be visualized on the CPOF screen. Once completed and visualized, the user could share the product in collaboration with other CPOF users. As a result, the skill hierarchy used the functional groups of Construct, Visualize, and Collaborate. Within each of these functional groups are tools, processes, and products related to the functions. This organization emphasizes the main functional components of CPOF with its military application implicit within those components.

Figure 2 illustrates the schematic architecture of CPOF skills produced by the task analysis. Accordingly, the CPOF system is characterized by four main functional groupings: Construct, Visualize, Collaborate, and System Basics. Each of these major pieces is further divided, to one degree or another, into the sub-pieces of Processes, Products, and Tools. "Processes" are the procedures for executing tasks (e.g., the steps for drawing a Main Supply Route). "Products" are the results of the procedures (e.g., the resulting Main Supply Route). "Tools" are the CPOF software features used in the procedures (e.g., the Graphics Palette).

As a tool to aid military decision-making, CPOF allows the user to visualize current and past battlespace information and to interactively share that information with other decision makers. Thus, "Visualize" and "Collaborate" are the primary CPOF functions in the schematic (i.e., Figure 2). Each of these primary functions contains products that allows for the execution of the specific function. For example, a Mapboard (a product under "Visualize") allows the CPOF user to view the terrain, battle graphics, Events, etc. that provide a picture of the current situation. The Visualize function and the Collaborate function are interdependent. That is, the CPOF products that are typically shared during collaboration are Visualization products. Likewise, the main purpose for Collaboration is to produce Visualization products.

The Construct function refers both to the generic capability of CPOF to construct virtual products and to the construction of specific hierarchical products for Visualization and Collaboration. The products contained in the Construct function represent the generic CPOF entities that are combined to form the products that allow for (primarily) Visualization. Thus, the Construct function can be viewed as subordinate to Visualize and Collaborate.

The final CPOF function represented in the schematic is System Basics. This function refers to the tools and processes needed for system operation and the interface. For example, one needs to know how to properly operate the mouse (e.g., left-clicking versus right-clicking) in

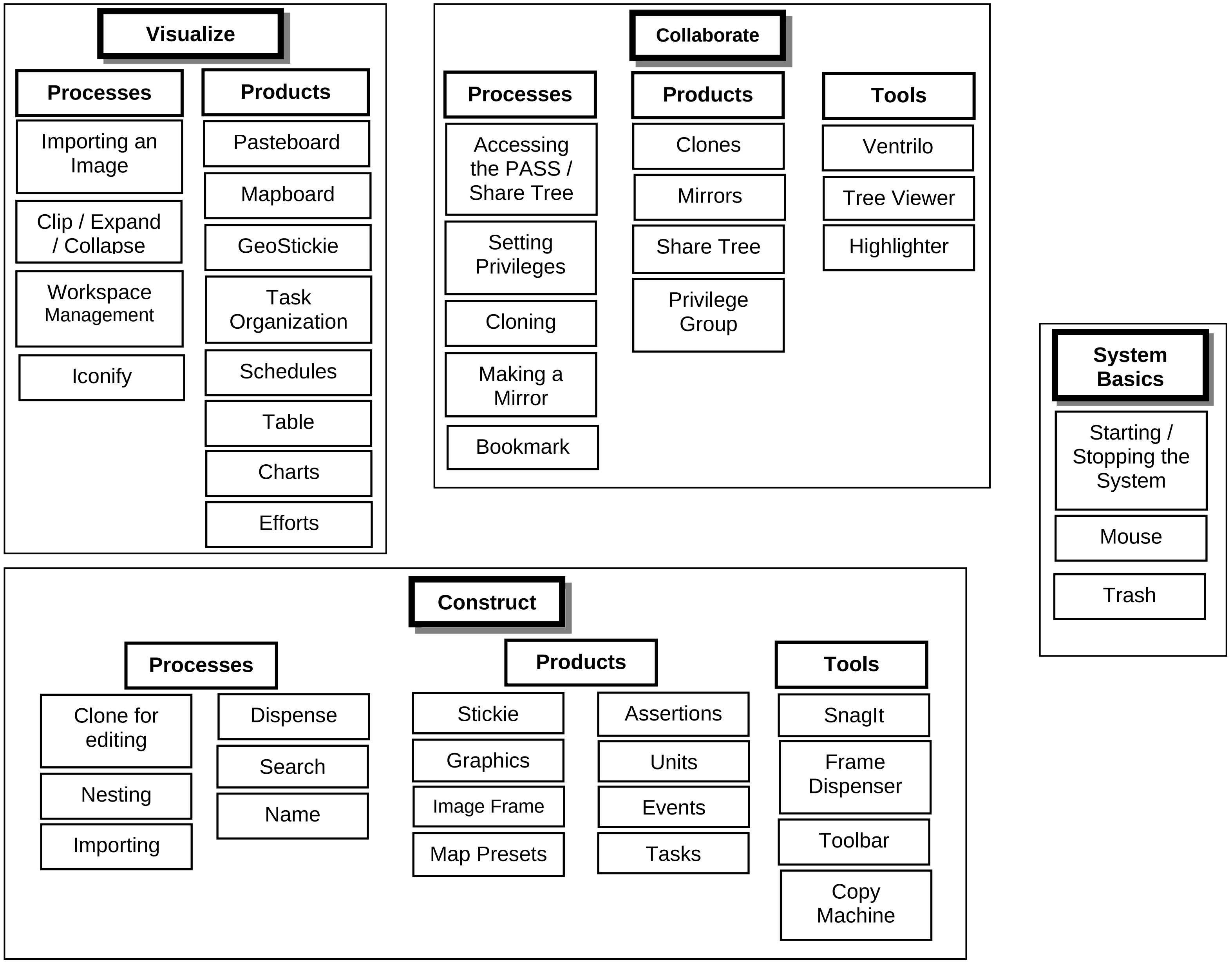

Figure 2. Schematic Architecture of Major CPOF Functional Capabilities

order to properly operate the system. System Basics also includes layout options that are common to all CPOF products.

It is important to note the relation between the CPOF schematic architecture shown in Figure 2 and the organization of the Critical Skills Document in the Appendix. Figure 2 focuses on the functional relations among CPOF elements while the Critical Skills Document focuses more on the training or knowledge overlaps. Thus, Figure 2 and the Critical Skills Document are not meant to directly mirror each other. For instance, “Workspace Management” is listed under “Visualize” in Figure 2 because it is functionally part of how one creates Visualizations of situations. However, it is part of the “System Basics” section of the Critical Skills Document because for training purposes it is part of the basic knowledge of the system and a foundation for aiding efficient learning. As another example, it makes more sense to cover the Frame Dispenser in System Basics in the Critical Skills Document rather than as part of “Construct” (as in Figure 2) even though the Frame Dispenser does play a large role in constructing CPOF products. The Frame Dispenser is involved in many CPOF procedures so, from a learning standpoint, it makes sense to have it appear early in the System Basics section.

In order to summarize the results, the critical CPOF skills that a learner needs to acquire are listed in Table 4. The critical skills are presented with some of the subgoals necessary to execute the critical skills. The list of sub-goals is not exhaustive, however, as the Critical Skills Document provides much more detail about each critical skill. Table 4 summarizes the CPOF critical skills identified in this research and provides an example of the types of subgoals identified for the critical skills. Table 4 can also be viewed as an advanced organizer for the Critical Skills Document.

Table 4
CPOF Critical Skills and Subgoals

	Critical Skill	Subgoals
System Basics	Starting the System	
	Stopping the System	
	Mouse (hardware): Clicking and Dragging	
	Frame Dispenser	
	Dispense	
	Name	
	Copy Machine	
	Toolbar	
	Iconify	
	Trash	
	Importing	
	Nesting	
	Search	
	Workspace Management	

Note. A blank line in the subgoal column means there is no subgoal identified.

Table 4 (continued)
CPOF Critical Skills and Subgoals

	Critical Skill	Subgoals
Construct	Create a Map	
	Rename a Map	
	Create a Stickie	
	Make a Clone for Editing an Item on a Map	
	Graphics	Creating Operational Graphics
		- Draw a check point
		- Draw a Main Supply Route
		- Draw a Boundary
		- Draw a Forward Operating Base
		- Draw an Objective
		- Draw air path
		- Find distance of air path
		Edit Operational Graphics
		- Change property (label, color, etc.) of a Graphic
		- Edit control points
		- Troubleshooting control points
		- Erase a graphic
		Dragging Graphics and Control Measures
	Create an assertion (e.g., Blue [friendly] assertion)	
	Create a Unit	Make Unit subordinate to another Unit
	Create an Event	
	Create a Task	
Visualize	Build a Pasteboard	Placing things on Pasteboard
	Moving and Zooming on Maps	Move on 2D Maps with Navigator
		- Move in small increments with Navigator compass
		- Go to grid location with Navigator box
		Zoom in/out on 2D Map
		Zoom in/out on 3D Map
		Name a GeoStickie
	Map Views	Make a preset
	Build Task Organization	
	Efforts	Create an Effort
		Put a Stickie in an Effort box
		Display Effort
		View Subefforts
	Schedules	
	Tables	Display data in columns
	Charts	Display Events on a timeline
	Import a Digital Image	
	Clip	
	Expand/Collapse	

Note. A blank line in the subgoal column means there is no subgoal identified.

Table 4 (continued)
CPOF Critical Skills and Subgoals

	Critical Skill	Subgoals
Collaborate	Sharing and Saving Products	
	Retrieving Shared Products	
	Tree Viewer	
	Setting Privileges	Adjust Privileges
		Adjust Effort Privileges
	Clones and Mirrors	Create a Clone
		Create a Mirror
		Access a Clone from Shared Products
		Access a Mirror from Shared Products
	Ventrilo: Communicate by voice	

Note. A blank line in the subgoal column means there is no subgoal identified.

Discussion and Recommendations

The goal for this research effort was to produce a conceptual framework for CPOF critical skills that can lead to the development of a training approach for CPOF. The essence of the conceptual framework was captured in a Critical Skills Document that included hierarchical structure among skills. The Critical Skills Document was the result of an expert knowledge elicitation process that focused on extracting subgoals from procedural steps (Catrambone, 1998). Given (a) the hierarchical nature of the Critical Skills Document, (b) the emphasis of the Critical Skills Document on abstracting procedures from specific examples, and (c) the emphasis of the Critical Skills Document on identifying similarities in procedures for different tasks, the Critical Skills Document provides a foundation for creating CPOF training. The structure within the Critical Skills Document allows one to organize training for the non-linear or seemingly "nonstructured" CPOF interface.

Even though the Critical Skills Document carries implications for training, it was not intended to be a training document that can be handed to a learner. Rather, it is a tool for helping someone to develop training and assessment materials and approaches. The Critical Skills Document identifies "what" needs to be instructed (and, presumably, assessed) but does not dictate "how" to do the instruction. The organization of the CPOF Critical Skills Document does provide a guide, though, to the order of instruction and the relations that should be emphasized. In addition, the schematic architecture (i.e., Figure 2) can serve as an advanced organizer to give people a reference of concepts as they learn CPOF skills.

The Critical Skills Document does not explicitly suggest an order in which CPOF tasks could be trained. However, the hierarchy uncovered by the task analysis process as well as procedural overlap among tasks could be leveraged to organize training. For instance, the task analysis process made it clear that Soldiers need to learn aspects of system basics (e.g., the Frame Dispenser) in order to effectively learn other tasks that depend on those basics (e.g., drawing graphical objects). Likewise, Soldiers need to construct a product before it can be visualized or used for collaboration. Therefore, Soldiers must learn how to construct at least some of the basic products (i.e., Construct Products) before training skills in the Visualize

Products and Collaborate Products levels. It is not the case, however, that *all* of the processes and products from these basic functional groups (i.e., System Basics and Construct) need to be trained before the processes and products in other functional groups. Given the overlap among CPOF critical skills and the complexity of some of the critical skills, it appears that the most logical training organization would include something similar to the following: (a) train some System Basics and Construct processes and products, (b) have students apply that knowledge in an operationally-relevant task, (c) introduce some Visualize and Collaborate processes and products that incorporate already learned elements, (d) have students apply that knowledge, and (e) continue to introduce and incorporate more complex processes and products from across the functional groups while reinforcing the training with application.

In general, the key to developing new training techniques for CPOF critical skills will be to provide instruction in a way that minimizes cognitive load while maximizing efficiency (Byrne, Catrambone, & Stasko, 1999; Gerjets et al, 2004; Mayer, 2005; Scheiter, Gerjets, & Catrambone, 2006; Sweller, 2005). While the Critical Skills Document does not specifically indicate *how* the CPOF skills are best trained, the structure and content of the Critical Skills Document hints that some training techniques better lend themselves to CPOF skills than other techniques. More specifically, training techniques that leverage the execution of subgoals and that illustrate overlapping CPOF procedures should most efficiently train CPOF skills. One such training technique is problem-based training in which learning occurs as the result of facilitated problem solving (Hmelo-Silver, 2004).

In problem-based training, the trainer provides learners with structured problems designed to develop a specific skill. The trainer does not necessarily provide information, but rather serves to facilitate the problem-solving process. The learners typically work in groups to solve the problem. The solution process requires the learners to develop learning strategies and to explore content knowledge. The important aspect of problem-based training is that each problem is structured around the use of a specific skill requirement (Hmelo-Silver, 2004). In the case of CPOF skills, problems should be based on the subgoals identified in the Critical Skills Document and should require learners to discover the overlap among tools and procedures. For example, a problem that requires learners to construct and locate a Stickie, an operational graphic, and a Unit on a map would reinforce an important CPOF subgoal (i.e., "locate") and would allow the learners to discover the similarities and distinctions among these three products.

While no direct evidence yet exists to support the assertion that problem-based training will be effective for CPOF skills, research on training other digital systems suggests that problem-based training was effective for complex systems like CPOF. For example, Childs, Blankebeckler, and colleagues (Childs, Blankenbeckler, & Dudley, 2001; Childs, Schaab, & Blankenbeckler, 2002) compared constructivist training techniques (e.g., problem-based training) to lecture-based training for the All Source Analysis System and the Advanced Field Artillery Tactical Data System. The results showed that problem-based training produced higher scores than lecture-based training on the performance-based practical exercises at the end of the digital skills courses. In addition, Childs et al. (2002) reported that problem-based training allowed for more material to be trained in less time without the perception of additional workload.

Of course, problem-based techniques might be only one tool for training CPOF skills based on the Critical Skills Document. The structure of the Critical Skills Document suggests that a mix of training techniques may be appropriate. For example, providing direct instruction on the System Basics should precede problem-based exercises in order to provide requisite system knowledge. From that point, a series of problems that focus on simple subgoals and procedural overlaps should be executed. Finally, more complex problems that require the construction and use of higher-level products can be addressed.

The Critical Skills Document presents a conceptual framework for CPOF digital skills based on a specific KE process. The best course of action is to obtain further empirical verification of the identified critical skills and the structure of the critical skills before much effort is devoted to developing a training approach based on the Critical Skills Document. It is important to once again note that the contents of the Critical Skills Document were not intended to be exhaustive. That is, the contents of this research product were limited to the applied knowledge of the chosen DEs and to version 3.0.2P2 of the CPOF system software. Additional research should extend the scope of the Critical Skills Document. Nevertheless, both the conceptual framework and the resulting training implications presented in this Research Report can be useful in developing a program of instruction for CPOF. Because CPOF currently has no official program of instruction, the Critical Skills Document can contribute to the development of formal CPOF training materials.

References

Anderson, J. R. (1983). *The architecture of cognition*. Cambridge, MA: Harvard University Press.

Anderson, J. R., Boyle, C.F., Farrell, R., & Reiser, B.J. (1987). Cognitive principles in the design of computer tutors. In P. Morris (Ed.), *Modeling cognition*. New York: Wiley.

Anzai, Y., & Simon, H. A. (1979). The theory of learning by doing. *Psychological Review, 86*(2), 124-140.

Byrne, M.D., Catrambone, R., & Stasko, J.T. (1999). Evaluating animations as student aids in learning computer algorithms. *Computers & Education, 33*, 253-278.

Catrambone, R. (1994). Improving examples to improve transfer to novel problems. *Memory & Cognition, 22*(5), 606-615.

Catrambone, R. (1995). Aiding subgoal learning: Effects on transfer. *Journal of Educational Psychology, 87*(1), 5-17.

Catrambone, R. (1996). Generalizing solution procedures learned from examples. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 22*(4), 1020-1031.

Catrambone, R. (1998). The subgoal learning model: Creating better examples so that students can solve novel problems. *Journal of Experimental Psychology: General. 127,* 355-376.

Chen, S. (2002). A cognitive model for non-linear learning in hypermedia programmes. *British Journal of Educational Technology, 33,* 449-460.

Childs, J. M., Blankenbeckler, P. N., & Dudley, M. G. (2001). *Digital skills training research* (ARI Contractor Report No. 2001-04). Alexandria, VA: U.S. Army Research Institute for the Behavioral and Social Sciences. (DTIC No. AD A391 790)

Childs, J.M., Schaab, B.B., Blankenbeckler, P.N., (2002). *Digital Skill Learning Using Constructivist Training Methods*. (Interservice/ Industry Training Simulation & Education Conference Proceedings). Orlando, FL: National Training Systems Association.

Dufresne, R. J., Gerace, W. J., Hardiman P. T., & Mestre, J. P. (1992). Constraining novices to perform expertlike problem analyses: Effects on a schema acquisition. *The Journal of Learning Sciences, 2,* 307-331.

Dunlosky, J., & Lipko, A. R. (2007). Metacomprehension: A brief history and how to improve its accuracy. *Current Directions in Psychological Science, 16*, 228-232.

Ellis, R. D., & Kurniawan, S. H. (2000). Increasing the usability of online information for older users: A case study in participatory design. *International Journal of Human-Computer Interaction, 12,* 263-276.

Farrell, I. H., & Moore, D. M. (2000). The effects of navigation tools on learners' achievement and attitude in a hypermedia environment. *Journal of Educational Technology Systems, 29*, 169-181.

Ford, N., & Ford, R. (1993). Towards a cognitive theory of information accessing: An empirical study. *Information Processing and Management, 29*, 569-585.

Gerjets, P., Scheiter, K., & Catrambone, R. (2004). Designing instructional examples to reduce intrinsic cognitive load: Molar versus modular presentation of solution procedures. *Instructional Science, 32,* 33-58.

Hmelo-Silver, C. E. (2004). Problem-Based Learning: What and How Do Students Learn? *Educational Psychology Review, 16*, 235 – 266.

Kalyuga, S., Chandler, P., & Sweller, J. (1999). Managing split-attention and redundancy in multimedia instruction. *Applied Cognitive Psychology, 13*(4), 351.

Kieras, D. E., & Bovair, S. (1986). The acquisition of procedures from text: A production-system analysis of transfer of training. *Journal of Memory and Language, 25*, 507-524.

Laird, J. E., Newell, A., & Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence, 33*(3), 1-64.

Larkin, J., McDermott, J., Simon, D. P., & Simon, H. A. (1980). Expert and novice performance in solving physics problems. *Science*, *208*, 1335-1342.

Mayer, R. E. (2005). Cognitive Theory of Multimedia Learning. In R. E. Mayer (Ed.), *The Cambridge Handbook of Multimedia Learning* (pp. 30-48). New York: Cambridge University Press.

Newell, A., & Simon, H. A. (1972). *Human problem solving*. Englewood Cliffs, NJ: Prentice Hall.

Reed, S. K., Dempster, A., & Ettinger, M. (1985). Usefulness of analogous solutions for solving algebra word problems. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 11*(1), 106-125.

Ross, B. (1989). Distinguishing types of superficial similarities: Different effects on the access and use of earlier problems. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 15*(3), 456-468.

Ross, B. H., & Kennedy, P. T. (1990). Generalizing from the use of earlier examples in problem solving. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 16*(1), 42-55.

Scheiter, K., Gerjets, P., & Catrambone, R. (2006). Making the abstract concrete: Visualizing mathematical solution procedures. *Computers in Human Behavior, 22*, 9-25.

Schraagen, J. M., Chipman, S. F., & Shalin, V. L. (eds.) (2000). *Cognitive task analysis*. Hillsdale, NJ: Erlbaum.

Sweller, J. (2005). Implications of Cognitive Load Theory for Multimedia Learning. In R. E. Mayer (Ed.), *The Cambridge Handbook of Multimedia Learning* (pp. 19-30). New York: Cambridge University Press.

Zhu, X., & Simon, H. A. (1987). Learning mathematics from examples and by doing. *Cognition and Instruction, 4*(3), 137-166.

Acronyms

CPOF	Command Post of the Future
DE	Domain Expert
KE	Knowledge Extraction
KEE	Knowledge Extraction Expert

APPENDIX

CPOF Critical Skills Document

The following Critical Skills Document is organized around subgoals (at different levels), steps (for achieving subgoals), and information relevant to CPOF, subgoals, and military issues .

Subgoals are in bold and are always preceded by a dash bullet (-).

Steps are also preceded by a dash bullet (-).

Information is always preceded by a filled circle bullet (•).

Indentation is used to indicate a hierarchy; for instance, the steps for achieving a particular subgoal will be indented relative to the subgoal statement.

Italics are used for emphasis or to highlight the name of something.

When CPOF concepts or procedures are mentioned before they are fully described, footnotes are given that reference the full description.

Using this Critical Skills Document

The aim of this document is to provide instructional designers and trainers with key information that should be conveyed to learners. This document, while organized in a way that attempts to combine related aspects of CPOF, does not specify *how* to develop training materials or what order topics should be covered. CPOF is a non-linear system and topics can be covered in a variety of orders--and some topics might be completely omitted--depending on the background of the learners, the learners' goals, the instructor's goals, and the time available.

CPOF is hierarchical; that is, a variety of procedures have similar goal structures. This document takes advantage of this overlap by making the subgoal and step structure of procedures explicit so that other procedures covered in this document with a similar structure can be learned more quickly and that, more importantly, procedures not covered in this document--and perhaps not covered in training--can be learned or inferred by the user.

Given CPOF's goal structure, an instructional designer or trainer would want to take advantage of this to sequence training to emphasize subgoal overlap so that learners can leverage earlier procedures when learning later ones. This will allow training to occur more efficiently because learners will be able to see the relationships to earlier procedures and thus be able to use previously-learned procedures to learn or infer new ones more quickly compared to a situation in which the overlapping subgoal structure was not made apparent.

For example, consider two tasks: Creating a unit and creating an event. If you examine their steps (p. A-10 to A-11) you will see a good deal of overlap. They both require getting something (a Palette) from the Frame Dispenser; they require clicking on the relevant tab; the desired object is dragged to the desktop; a grid is specified; various details about the to-be-created object can be specified. There are also differences such as the nature of the details to be specified (for a unit one will likely specify whether it is friendly or not; for an event one might specify whether there was small arms fire). The structure of the procedures for creating these two objects--units and events--are very similar and thus it is to a learner's advantage to have this structural overlap emphasized so that after learning the procedure for creating one of these objects, the learner should be able to relatively quickly learn (or infer, if necessary) the procedure for the second.

CPOF CRITICAL SKILLS

SYSTEM BASICS: FUNDAMENTAL PROCEDURES AND FACTS

– Starting the System

- With 3 screen system, the leftmost screen is the Windows screen
- Running across bottom of the screens is the Windows task bar (a blue bar)
- On the right side of the bar will be some icons including a green star
  - Right click on the star
  - Options appear
  - Move pointer to “Start” option (others are stop, profile, about..., and exit)
  - Various suboptions appear (the applications of CPOF: 2D, 3D, Maps, voice, and All)
  - Left click on “All” and system will start up
- When system is started, the three screens will show whatever they were showing the last time it was stopped unless the system was specifically cleared

– Stopping the System

- Any one of the four systems can be stopped individually via the “Stop” option
  - Right click on green star in windows task bar
  - In the options that appear, move pointer to “Stop” option
  - Various suboptions appear (the applications of CPOF: 2D, 3D, Maps, voice, and all)
  - Left click on desired suboption

– Mouse (hardware): Clicking, and Dragging

  - Left click: to move things or select
  - Right click: to access sub menus (including delete/remove options) or additional choices
  - When dragging things around, generally want to grab at the tab and/or be sure that the “moving” symbol appears when pointer is on/over the thing (object, window, etc) that the user wants to move

– Frame Dispenser

- Like a toolbox
  - To make Frame Dispenser appear: right click anywhere on desktop
  - Items are obtained from Frame Dispenser by left clicking and dragging onto desktop
- Tools/Objects from Frame Dispenser include: Maps, Toolbar, Graphics Palette, Task Palette, Effort Palette, Unit/Event Palette

– Dispense

- Is the act of moving (dragging) an Event, Task, Unit, etc. from its Palette to the desired item (Map, Table, etc)

– Name

- Is the process of labeling the Efforts, Pasteboards, etc. created

– Copy Machine

- Get from Frame Dispenser

- Can Clone[1] something, drag it to copy machine, then click “make copy” and then can drag result wherever the user wants
- This process severs link to original and can now be edited regardless of Privileges[2] of original

– Toolbar

- Obtained from Frame Dispenser
- Some useful tools from Toolbar include:
  - Flashlight: allows the user to draw on a Map and then the drawing fades away after a few seconds.
  - Text tool: select the text tool, move cursor to Map, left click, and start typing.
  - Highlighter (looks like a yellow highlighter)
    - Can be used to put highlighting around text (only text) on a Map
    - When the user clicks on highlighter, a set of pastel options will appear
    - Click on a color and then move highlighter to Map
    - Place it in the text (highlighter symbol changes to directional arrows) and then click on text and it will become highlighted with the selected color

– Iconify

- To iconify something is to turn that thing into an icon (perhaps because the user wants to reduce clutter on the desktop)
- Pasteboards, Maps, other objects have a tab at top with various symbols for info/options such as the object's name, Privilege info, etc.
  - If the user moves the mouse over one of these symbols, the name of the symbol or the info the symbol represents will be shown.
  - One of the symbols (squareish with an arrow pointing into it) is the “iconify” function
  - Single left clicking on it will turn that object into an icon
  - Double clicking on the icon will expand it back to the original object

– Trash

- To trash something (box, window, etc), there is usually a symbol on the top tab on the right (looks like the eye of a hurricane)
  - Left click on it and the object goes to Trash
  - Alternatively, can drag the object to the trash can which is a eye of hurricane symbol on desktop and is called “Trash”

– Importing

- Is like copying data from a central repository
  - Locate data (file, manifest, image, etc) in Shared Products[3] (should be a Subeffort)
  - Control, drag each Subeffort to the workspace

– Nesting

- Is like sticking an object to the front of something

[1] Clone is described in the Construct section on page A-6
[2] Privileges are described in the Collaborate section on page A-12
[3] Shared Products are described in the Collaborate section on page A-11

- In general, nesting means to associate one subordinate product with another
  - When the user drags one object onto another, a nesting icon usually appears
  - Click on the nesting icon fairly quickly because it stays visible for around 5 seconds
    - For example, nesting a Map to the front of a Pasteboard means that if the user clicks elsewhere on the Pasteboard[4] the Map will stay out front; otherwise, if the user clicked elsewhere the Map would end up behind the Pasteboard
- The user can nest a Pasteboard (and other things) to a "main" Pasteboard
  - To do this, must first configure the main Pasteboard for automatic layout
    - With the cursor in the main Pasteboard (not on a Map), right click to see the automatic layout option and left click on that option
    - Move the top tab line of the to-be-subordinate Pasteboard (or Map or other object) to the 2nd banner line (the "banner bar") of main Pasteboard and then nest it
    - A button for the subordinate Pasteboard will now appear as a tab in 2nd banner line of main Pasteboard

– Search

- Is a function to help find data using key words, just as search in MS Office and Internet-based systems

– Workspace Management

- Keep center screen as primary work space
- Center screen will typically have the main Pasteboard and Map (2D Map)
- Keep Tools and Palettes on right screen
- Iconify Tools and Palettes that are not being used much at the moment to create desktop space
- Maximize what is needed when it is needed

CONSTRUCT

– Create a Map

  - Get Map from Frame Dispenser
  - Drag and drop it on Pasteboard
- Maps can be placed directly on the desktop but normally they are placed on Pasteboards

– Rename a Map

  - Go to Map tab
  - Right click
  - Select rename
  - Use BACKSPACE key to remove current name and put in the role name (e.g., 1-8cav)
  - Hit the ENTER key

– Create a Stickie

- Like a post-it note

[4] Pasteboard is described in the Visualize section on page A-14

- Go to Frame Dispenser and get Stickie (called “Untitled Stickie”; this is not a GeoStickie)
- Left click and drag to desktop
- Type desired note/info on to Stickie
- Drag Stickie to desired location
- Then Nest it

– Make a Clone for Editing an Item on a Map

- If item is on Map and the user wants to edit that item, one way to do it is to make a Clone of that item
- The clone can be thought of as a temporary copy of the original that the user can edit and then the user can throw away the copy once the changes have been made
  - Press and hold the SHIFT key
  - Left click on item on Map and drag it to the desktop
  - Typically, some sort of box appears that allows the user to specify features of the item
    - This includes specifying, for example, the type of Event, so that the user can drag **any** Event to Map and then adjust the type using the Clone [the box that appeared on the desktop]
    - Note that if the user changes the grid (location) of the item, once the user is done making changes, the user might no longer see the item on the current Map if the chosen grid is not covered by the current Map
  - As the user alters those features, the appearance of the item on the Map will change
  - When the user is done making changes, it is fine to Trash the Clone on the desktop because all the information/changes remains in the item on the Map
    - The user can verify this by cloning the item again (dragging item to desktop) and looking at the values

– Graphics

- There are three types of operational graphics: Point, Linear, and Area symbols.
- An important difference among these graphic types are the number of “control points” that make them up.
- Control points are the minimum number of points needed to specify the object being drawn (i.e., if the “dots are connected” then the object is drawn)
  - For example:
    - A straight line (a “linear” graphic) needs two control points (beginning and end) to specify its location
    - A rectangle (an “area” graphic) needs four control points (the four corners) to specify its location

– Create Operational Graphics

- Operational graphics typically have a specific Map location to which they must be assigned
  - First, get Toolbar and Graphics Palette from Frame Dispenser
  - Go to Map location that is roughly the right location for the operational graphic the user is about to create

- Put the Graphic somewhere on the Map and *then* specify the grid(s) for it (see section on editing operational graphics)
  - If the grid(s) for the operational graphic is not on the current Map, the graphic will seem to disappear because it has now moved to the relevant grid(s)
  - This is why it is best to first have up a Map of the right general areas for the graphic
- Go to the Graphics Palette and select desired type of graphic from the *graphic type* drop down box
- Go to *Toolbar* and left click the Ink tool (looks like a pencil/pen)
  - Cursor now looks like pencil
- Go to Map and start drawing the graphic
- Generally speaking, the user left clicks once to start drawing and then move cursor until the user reaches the point where the user needs to change direction; at this point the user clicks again, etc
- Each click will produce a control point
- For instance, to draw a straight line the user would click once to start line and then click at end of line

• Point graphics
  • Examples are a Check Point or Coordination Point

– Draw a Check Point
- Go to Graphics Palette and select Check Point from *graphic type* drop down box
- Select ink tool from Toolbar
- On 2D Map, left click and release; this creates the Check Point
- Get rid of Ink tool by clicking arrow in Toolbox to go back to cursor mode

• Linear graphics
  • Examples are a main supply route (MSR) and a boundary

– Draw a Main Supply Route (MSR)
  • Planning a route usually means following a road if one exists
- Select “main supply route” from graphics palette
- Select Ink tool from toolbar
- Go to 2D Map (or 3D) and do a series of left clicks to create control points to build path
- At last control point do double left click
- “MSR” will appear on the route

– Draw a boundary
- Select boundary from Graphics Palette
- Select Ink tool from Toolbar
- Draw boundary
- Default is brigade boundary (the user will see an x on it)
- Must go to 3D Map to change the boundary type

• Area graphics

- Examples are a forward operating base (FOB), objective (OBJ), air paths, and Named Area of Interest (NAI)

– Draw a Forward Operating Base (FOB)

- NOTE: a FOB is always a closed geometric shape defined by 3 or more grid locations

- Count the number of needed grids to specify FOB
- The number of grids will determine the number of **control points** that define the FOB (number of grids = number of control points)
- Go to Graphics Palette and select General Area from Graphic Type drop down box
- Select Ink tool from Toolbar
- On 2D Map, left click and release; this creates a control point
- Repeat for each control point
- Double click (rather than single click) after last control point
- Get rid of Ink tool by clicking arrow in Toolbox to go back to cursor mode

– Draw an Objective (OBJ)

- Select “Objective” from Graphics Palette
- Select Ink tool from Toolbar
- Create control points
- Double left click on the last point
- Left click on 3D Map
- Right click on control points to give them desired grid locations

– Draw air path

- Air path can be built ONLY on 3D; so look at Graphics Palette on 3D screen (this palette is present automatically on 3D screen)
- Air paths are different from other graphics in that the user can not edit specific grid locations
- Rather, use GeoStickies[5] at specific grid locations to guide the beginning and end points (and midway points if desired) of the air path
- Single left click on air path icon
- Move cursor to Map and single left click at start or end point
  - An air control point will appear
- Single left click and release at each turning point on air path
- Keep doing this until the end point is reached
- Final point is still a single left click and release
- Then go to Toolbar to obtain a “normal” cursor

– Find distance of air path

- Not straight line distance, but total distance

- Single left click on air path to activate it (3D Map only)
- Look in Ink Properties window in bottom right corner of 3D screen and scroll down in that window and distance will be displayed

[5] Geostickies are described in the Visualize section on page A-15

– Edit Operational Graphics

- To edit graphic on 3D Map, the user must first connect the 2D Map with the 3D Map
- To connect 2D Map to 3D Map, click on “connect” button on navigator on 2D Map
- Once this button is clicked, “connect” becomes “connected”

  – Change property (label, color, etc) of a Graphic

  - There are three ways; one fairly standard way is:
  - On 3D Map, single left click the object
  - **Ink Properties box** will appear in the lower right corner of 3D screen
  - This box shows various properties of the Graphic and will often contain drop down menus for changing property values or fields for typing in values

  – Edit control points for a Graphic

  - This must be done on 3D Map
  - On 3D Map the user must go into *point edit mode*
    - – Do that by double left clicking on the object of interest (if the object is an area Graphic, then click on one of the lines of the object rather than in its center)
  - When the user does this, the control points (or point, if it is a point Graphic) will appear
  - Note that the control points have grid locations associated with them, but these locations might have been more or less arbitrarily chosen when the user initially drew the object
  - Choose a control point by right clicking on point
    - – Note that a control point will turn yellow when pointer is over it, indicating that it is ready to be chosen
  - Multiple options might appear when the user right clicks; choose the option “edit point coordinates” by left clicking on it
  - “edit point” box appears
  - Use BACKSPACE key to remove the current grid and type in desired grid
  - Click ok
  - Go to next point (if there is more that one point)
  - Go clockwise or counterclockwise and repeat process
  - Note that the user can move mouse over each point (i.e., “hover” over that point) to see the grid numbers for that control point

  – Troubleshooting control points for Graphics

  - If the user specifies control points in an order that does not match the grid order, the resulting Graphic will be oddly-shaped (e.g., a bow tie rather than a rectangle)
  - To fix this the user must work on 3D Map
  - Hover over a control point to get the grid info for that point
  - Identify the control point that corresponds to the first specified grid location
  - Identify the second control point
  - Then move the first control point to fix the geometric problem
  - Do this by left click drag and drop
  - It is not important at this point to place the control point at a precise location; purpose here is to just make Graphic have correct general shape
  - Now assign those two control points with the correct grid locations

– Erase a Graphic

- If the user draws any Graphic and then wants to erase it, do the following:
- Get Eraser from Toolbar
- Run the eraser through any line in the Graphic and the Graphic will then disappear

– Dragging Graphics and Control Measures

- • Sometimes the user will want to make a group of Graphics part of an Effort[6]
- To do this the user will need to drag that group of Graphics to that Effort
- The user can select a set of Graphics (as opposed to moving them one at a time) by
  - Left clicking and holding
  - Dragging cursor across the desired Graphics
  - Releasing click
  - Now the Graphics are selected and the user can click and hold on one of them and they will all move when the user begins dragging
- Once the user has selected the Graphics and begins to drag them, if the user decides he or she did not want to drag them, DO NOT STOP while on the Map, otherwise if the user lets go of the Graphics, they will stop in new locations and there is no way to undo that placement
  - Instead, continue dragging them to the desktop
  - Release them (let up on mouse clicker)
  - Select the graphical items again (the ones the user dragged to desktop) and drag them back to Map and release; they will return to their original positions on Map as long as the user did not change the grid coordinates

– Create an Assertion (e.g., blue [friendly] assertion)

- Right click anywhere on 3D Map
- Submenu appears
- Select type of Assertion from submenu
- Left click on it
- Appropriate symbol appears
- Go to Ink Properties box to adjust values

– Create a Unit

- Get Unit/Event Palette from Frame Dispenser
- Click on Unit tab (if not already the selected tab)
- Drag any Unit to desktop
- Use BACKSPACE key to remove "untitled" and give it name (e.g., 2BCT)
- Type grid coordinates into grid coordinate field
- Select type of Unit (e.g., infantry)
- Use drop down boxes to select features (e.g., blue, friendly, brigade)
- Drag the window to the 2D Map anywhere
  - It might disappear if grid location for the Unit is not on current Map, but if the user goes to Map with that grid location on it, the Unit will be displayed there

• Can make several of these and Clone each on to Map

[6] Efforts are described in the Visualize section on page A-17

- **Make a Unit subordinate to another Unit (also true for Events and Tasks)**
  - Decide on parent
  - Clone parent and drag to desktop
  - See the box “contain elements here”
  - Clone each desired child to desktop
  - Drag each Clone into the “contain elements here” box of parent
  - **If the user wants to make changes to a child**
    - Clone child and drag to desktop (the original stays in box)
    - Make changes, then trash the Clone

- **Create an Event**
  - Get Unit/Event Palette from Frame Dispenser
  - Click on Event tab (if not already the selected tab)
  - Drag any Event to desktop
  - In resulting box, fill in information (title which is usually date and time)
  - Add info in drop down boxes
    - e.g. in “Type” the user might select “small arms fire (SAF)”
  - Type grid in the GridCoords box
  - In the Comments field, type summary of Event itself (in user's own words)
  - Drag and drop Event box to the 2D Map anywhere
    - It might disappear if grid location for the Event is not on current Map, but if the user goes to Map with that grid location on it, the Event will be displayed there

- **Create a Task**
  - Choose a Task from Task Palette
    - Every Task has a number by default
  - The user can give it a name (e.g., Operation Hammer)
  - Comments box can be anything such as mission statement
  - Clone the Task and drag it to a location on Map
  - Create a Map with various features, Efforts, etc.
  - Drag to the “contain elements” box of Task
  - Clone the Task (at any point) to the interior portion of the scratch schedule
    - A scratch schedule is a temporary schedule that allows the user to try out different scenarios
    - The scratch schedule does not share its information with a master schedule unless the user explicitly moves the information to the master schedule

COLLABORATE

- **Sharing and Saving Products**
  - About 200 clients can be using CPOF simultaneously
  - The Shared Products list is used to share the user's work with others
  - All data saved on server, not locally
  - The user can share things such as Pasteboards (with multiple tabs), Efforts, Tables, etc
  - Sharing is done primarily through the **Shared Products** list
  - This is a folder/file structure on the server that people contribute to

- Different organizations/Units using CPOF might organize their folder/file structures differently
- The folder/file structure created by a particular Unit or organization is called a *Share Tree*
- If the user drags a Pasteboard or other product to the Shared Product list, then it is off the user's desktop and saved on the server (in the Shared Products list)

– Getting Shared Products

- Get Shared Products from Frame Dispenser
  - – EXAMPLE: Suppose the user wants to get an Effort from a particular Pasteboard named BTL CPT COP
    - - Go to BTL CPT COP Pasteboard in Shared Products list
    - - Grab and drop the Effort onto Map
    - - Effort will show up in Effort list

- If something is dragged from shared product list to desktop without holding the SHIFT key, then the thing on the desktop is a Mirror and the user can not move among tabs (if there are any)
- A mirror is basically a frozen snapshot of one page of an object and nothing changes unless the owner makes changes

– Tree Viewer

- Allows the user to examine work products (Maps, Pasteboards, plans, orders, etc.) In desired hierarchy
- The icons at the top of the Tree viewer allow the user to change the hierarchy of the information being displayed
  - – Drag out a Tree Viewer from the Frame Dispenser or right click the workspace desktop and select browse workspace
  - – The Tree Viewer can be adjusted to display and organize work products
  - – Drag out a Pasteboard and nest Frames or elements in the Pasteboard; the Pasteboard will appear in the Tree viewer with (+) in front of it
  - – Click the (+) to expand and view the products in the tree like expanding and viewing folders within Microsoft Explorer

– Setting Privileges

- When the user puts a Clone on Shared Products list, the Clone (that someone else accesses) will have no Privileges other than to allow the person to view it
- If the user wants other people to make changes to the Clone, the user would have to adjust the Privileges prior to putting that Clone on the Shared Products list
- Generally it is more efficient to change the Privileges of the object before cloning it and dragging the Clone to the Shared Products list
  - **– To adjust Privileges**
    - - Right click on eyeball icon (Frame Privilege editor)
    - - Open Privilege editor
    - - Right click in desired access box (view and/or content)
    - - Role names come up. Select the role by left clicking
    - - Trash the Privilege editor

– To adjust Privileges for an Effort

- Drag the relevant Effort to desktop on far right
- In field with the blue man symbol and Effort name; right click in that field and choose option to change Privilege
- Privileges from Maps, Pasteboards, etc. do not filter to Efforts; must be done for each Effort

– Clones and Mirrors

- If the user creates a **Clone** of something (e.g., a Pasteboard with a Map on it, a Table, an Effort) and put it on the Shared Products list:
- Another person can get that Clone from the Shared Products list and put it on their desktop
- That other person can view the Clone independently of how the user is viewing it
- That is, the other person can be at a different level of zoom, they could be on a different tab of the Pasteboard, they could be scrolled to a different part of a Table
- If the user (the owner) makes changes, those changes appear on the Clone
- If the user creates a Clone with appropriate Privileges, other people can make changes to their copy and those changes will appear on the user's copy

– To make a Clone

- Press and hold the SHIFT key
- Left click on tab and drag the item to be Cloned to either a new spot on the desktop or to the appropriate folder on the Shared Products list
- ****one key exception****
  - If the user wants to Clone a Map that is currently sitting on a Pasteboard, the user must Clone JUST the Map and not the Pasteboard
  - If the user Clones the Pasteboard on which the Map is nested, the resulting Clone (Pasteboard and Map) will be a Mirror
  - It is just the way the software is

- There is *another type of cloning* in which the user might Clone something from a Map--such as an Event--in which the Cloned item becomes a window on the user's desktop
  - In this window the user can make changes and when the user is done the thing on the Map reflects those changes
  - This type of cloning is discussed later in this document

– To make a Mirror

- Press and hold control
- Left click and drag to be mirrored to appropriate folder on Shared Products list

- If the user creates a *Mirror* of something and puts it on the Shared Products list
- Another person can get that Mirror from the Shared Products list and put it on their desktop
- What that other person will see will match exactly what the user sees
- If the user (the owner) moves around the object (e.g., go to a particular zoom on a Map), the other person will see the same thing
- If the user (the owner) make changes, those changes appear on the mirrored item

        - The other person can not make any changes to the mirrored item

– To Access a Clone from Shared Products List

– Find it in the Shared Products list
– Put cursor on product
– Press and hold the SHIFT key
– Left click and drag to desktop

– To Access a Mirror from Shared Products List

– Find it in the Shared Products list
– Put cursor on product
– Press and hold control
– Left click and drag to desktop

- The Clone and Mirror processes support varied aspects of decision making
  - Cloning is most appropriate for exchanges when the most current information or changes are critical planning or immediate tactical decision making
  - Mirroring might be most appropriate to support update briefings
    - The briefing HQ will need to use and control the display to assist the briefed HQ or commander in understanding an event or action or developing awareness/understanding of an event or action that occurred

– Ventrilo: To communicate by voice

- *Ventrilo* provides synchronous voice collaboration via VoIP (voice over Internet Protocol)
  – To use Ventrilo:
    - Right click the CPOF Star in the task bar and select Start -> Voice
    - Double click a channel name to talk and listen
- Ventrilo includes comment and chat capabilities, similar to Internet functions

VISUALIZE

– Build a Pasteboard

- A pasteboard is like a digital corkboard
  - It provides a “surface” on which to organize and manage information
  - The user can place Maps and other objects on a Pasteboard
  – Go to Frame Dispenser and drag and drop a Pasteboard on to desktop
  - A Pasteboard can be laid out in manual or automatic mode
    - Manual layout means things stick where the user places them
    - In Automatic layout, a *products bar* appears with tabs resembling buttons for each of the items on the pasteboard and with the Frame tab controls for the active item at the far right end of the bar

– Placing things on Pasteboard

– To place things on Pasteboard, the user typically just drags items to it and drop them there

- *2D and 3D screen issues*
  - There are tools across the top toolbar; some relate to 3D Map only, some relate to 2D Map
  - Some relate to zooming and direction and shading and orienting

– Moving and Zooming on Maps

- The *Navigator* is the graphic with the compass, the bar for typing grid info, and with the presets below it
- It always appears on 2D Map when the user brings up a 2D Map
- The Navigator allows the user to move to any location on 2D Map

– Two main ways to use Navigator to move around in 2D Map:

– To move in small increments with Navigator:

– Click on compass directions on Navigator

– To go to a grid location via the Navigator

– Go to Navigator and click in bar below the compass
– Grid info in that bar will disappear
– Type in new grid info (do not use spaces)
– Hit the ENTER key

- When the user moves to a new location with Navigator:
  - A *GeoStickie* (a thumbtack) appears at grid location and the user is taken to a Map view that has that location centered on it
    - It is a light blue circle and next to it will be the word "untitled"
    - Note that a GeoStickie is visible at any zoom level
    - A GeoStickie could be used for many things such as marking beginning and end points of some graphical object the user will be adding to a Map
      - If the user hovers on GeoStickie, a pop up box will appear telling the user the grid coordinates
      - Can remove a GeoStickie by right clicking and selecting "remove" or the user can drag to trash

– To name a GeoStickie

- "Clone" it and drag to desktop
- Press and hold the SHIFT key
- Left click on GeoStickie and drag it to desktop
- Type in name in the window that appears
- Trash the Clone
- Name of GeoStickie will show on Map

– To zoom in on 2D Map

- Decide the area on which to zoom in
  - The smaller the area selected, the greater the degree of zoom
  - Need to crop small area to get to imagery view
  - Otherwise the user will stay at more of a non-imagery view (looks more like a paper map)

- Move pointer to desired location
- Press and hold alt
- Left click and drag down to right
- Release alt key and mouse

– To zoom out on 2D Map

- One approach is to right click on the Map and the select the zoom out option
- Another approach is to click the minus symbol on the Navigator

– One of the ways to move around in 3D Map:

- Hold down the SHIFT key and left click and hold
- Hand shows up on screen; drag and release and movement will occur

– To zoom in and out on 3D Map

- Place Cursor on 3D Map and then use the scroll wheel to zoom (must depress the scroll wheel while rolling it)

– To create a distance tool

- Must be done on 3D Map
- Right click anywhere on 3D Map
- Select "create distance tool"
- Distance tool appears on Map

– Map Views

- A *Preset* is a bookmark that will allow the user to go to a Map quickly
- Presets are listed in the Navigator window right under the grid location bar
- Preset remembers specific point on ground and what zoom level; must specify both when creating preset

– To make a preset:

- Usually good idea to first click on Baghdad preset
  - This brings the user to a type of Map and zoom that is a good start point
  - If the user went right to the grid location bar to type in grid, the resulting Map/image will be the same type of Map/image and zoom as last preset which might not be what the user wants
- Double left click on "2D view" tab on navigator (must be 2D tab, 3D tab won't do anything)
- This creates a new untitled preset tab (in the list of presets in the navigator window) with "untitled" highlighted
- Hit BACKSPACE key to remove "untitled" and then type in name (e.g., Mosul)
- Hit the ENTER key
- Now have preset

– Task Organization

- Units, Events, and Tasks are typically items that the user builds on the desktop before dragging them on to the Map although the user can also first drag them to the Map and then drag a Clone from Map to desktop

– To build a Task organization

- A Task organization is really a hierarchy; a relation among Army Units
  - In Frame Dispenser, pull out Task organization Pasteboard
  - Drag Unit window (if it contains children it is a Task organization) onto the Task organization Pasteboard
  - Drag or Clone the Task organization to the left portion of window of a scratch schedule
  - If "affects schedule hierarchy" field of Task organization window was not populated, the subordinate elements did not show up in the left windowpane
  - So, Clone the subordinate Units (children) in the Task organization (in the "affects box") into the "affects schedule hierarchy" box

– Efforts

- An Effort can effectively be considered a Work Folder
- Each Map has an Effort List that is always present and shows all the Efforts associated with that Map
- Note that items from geographically different areas can be put into the same Effort

– To create an Effort

- Get *Effort palette* from Frame Dispenser
- In Effort palette, drag out *Effort*
- Left click on title of Effort box and type in the desired title
- Notice box "contain elements here"
  - Left click and then grab each desired item from Map and drag into the "contain elements" box
  - Alternatively can left click and drag across all the items and they will get highlighted and the user can drag them all to Effort window
- Dragging items to Effort window will preserve them for that Effort
- Drag the named Effort and drop it into Effort List of the Map

– To put a Stickie in an Effort box

- Stickie must be nested into the Effort box
  - – This is true even if the Stickie was already nested on Map
- Drag and drop Stickie into "contain elements" box
- Nest symbol will appear
- Click on the nest symbol

– In Effort List, the user can turn on or off an Effort (operational Graphics, Units, etc) by double clicking on the name of the Effort

- This will cause items in the Effort to appear or disappear from Map

- An Effort can contain Subefforts

– To see Subefforts

- Double click on a particular Effort and the Subefforts will appear

– Schedules

- – Master schedule is accessed from Frame Dispenser and shows all Tasks for all Units within the organization
  - Subset master schedule is used to focus on specific sub-Units within an organization
  - Scratch schedule is used for planning new operations
  - Tasks are created to define the what, where, why, and how by selecting from the *Task palette* and *Map*
  - The who and when for each Task are created when the Task is inserted into the *schedule*

– Tables are used to display data in columns

- – Drag desired data into tables
- – Click on data cell in table to mark data or to drill down

– Charts are used to display Events on a timeline

- – Drag an Event chart from the Frame Dispenser
- – Use control drag to move Events from an Effort to the Event chart

– Importing a Digital Image

- – Go to print icon; photo printing wizard displays
- – Select next; next again; make sure *Snagit 7* is default printer; next again; next again; finish
- – Crop excess white margin
- – Image->rotate 270 clockwise
- – Click on finish web (green checkmark)
  - - Image is now a .jpg file on CPOF
- – Go to Frame Dispenser
- – Drag out image overlay table and an image Pasteboard
- – The image overlay table is where the image landed
  - - Scroll to bottom to find image
  - - The user will see the user's roll name
  - - Left click hold and drag and drop on to image Pasteboard
- – Single left click on flap on that Pasteboard
- – Click icon image name under “fetch image” heading and the picture will display
- – Close the flap
- – Give name
- – Grab the tab and line it up with the banner bar (that shows all the other tabs) and release
- – The picture is now a tab on the banner bar
- Be sure to establish a naming convention to find and manage images (users typically accumulate a large number of images)

– Clip is used to edit or resize a Frame

- – Drag the handles on the edges of the Frame to clip (crop) and unclip the Frame.
- – Move the cursor over the left column of a Frame to display the stretch and squeeze handles.
- – Click and drag one of the side handles to stretch or squeeze the column horizontally.
- – Click and drag the top or bottom handle to stretch or squeeze the Frame vertically.

- Frames with multiple internal regions (like schedules or charts) have internal handles that allow the user to expand those regions.

– Expand / Collapse

- Expand / collapse icon is used to access icons in the Pasteboard's Frame tab
- Click the expand/collapse “<“ or “>“ icon at the top of the Frame tab
- Expand / collapse arrow is used to display contents or close boxes