(2)

SOFTWARE USABILITY ASSESSMENT GUIDE

Prepared by

PRC Inc.

&

American Systems Corporation

CONTRACT NUMBER MDA 903-88-D-0019

DELIVERY ORDER NUMBER 0040

CDRL A002 & CDRL A003

MANPRINT

TECHNICAL INTEGRATION DIVISION

U.S. ARMY OPERATIONAL EVALUATION COMMAND

AMENDED FINAL VERSION
APRIL 11, 1993

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE 11 APR 93	3. REPORT TYPE AND DATES COVERED Final Report

4. TITLE AND SUBTITLE Software Usability Assessment Guide	5. FUNDING NUMBERS C MDA 903-88-D-0019 TA 0040
6. AUTHOR(S) BISHOP, Glade M.; MOFFETT, Raymond J.; LICKISS, Rebecca; and BOURNE, William	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) PRC Inc. 4401 Ford Avenue Suite 420 Alexandria, VA 22302 American Systems Corp. 14200 Park Meadow Drive Chantilly, VA 22021	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) U.S. Army Operational Evaluation Command 4501 Ford Avenue (Park Center IV) Alexandria, Virginia 22302-1458	10. SPONSORING/MONITORING AGENCY REPORT NUMBER U-3328

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT Unlimited Availability to the Public	12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

This guide shows independent operational evaluators and operational testers how to measure the usability (or User Friendliness) of computer software components that require human interaction. It shows how to measure the effectiveness and efficiency of human interactions with the software components of any weapon, equipment, computer, or information mission area system. The methodology in the guide can be applied during any phase of the Acquisition Process. It also addresses the inherent trainability of software components. It covers long- and short-term planning, defining a system's software interfaces, preparation of questionnaires and other data collection instruments, data analysis, and the development of Usability Profiles to describe all the software interfaces of a system. It contains lists of questionnaire and checklist items, and examples of questionnaires, used to gather data on User Friendliness and human factors (or MANPRINT).

14. SUBJECT TERMS MANPRINT, Software Usability, User Friendliness, Operational Test and Evaluation (OT&E), User Testing, Human Factors, User Interface, Software Interface	15. NUMBER OF PAGES 110
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL

SOFTWARE USABILITY ASSESSMENT GUIDE

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

THE METHODOLOGY OF THIS GUIDE AT A GLANCE

Define the human-software interfaces of the system

Para. 1.6.2   Para. 2.3.2   Table 2.1   P. A-1

List all the human-software interactions (i.e., task elements) to be done at each interface

Para. 1.7.1   Pp. C-1 thru C-4

Plan and develop data collection instruments

Para. 2.2   Para. 2.3.4   App B, all   App C, all
App D, all   App E, all   App F, all

Get direct performance measurements from real users

Para. 1.7.4   Para. 2.3.5   Pp. A-1 thru A-6   App B, all

Get indirect software usability reports from real users

Para. 2.3.5   Pp. A-6 thru A-10   App B, all

Develop Usability Profiles for comparisons across:

Software Interfaces
Successive Assessments
Different Systems

Para. 2.3.6   Para. 2.3.7   Para. 2.4
Para. 3.1 thru 3.5   P. A-5   App G, all

CHAPTER 1
ASSESSING THE USABILITY OF SOFTWARE SYSTEMS

1.1 **WHO SHOULD USE THIS GUIDE?** This Guide is written for the Operational Evaluator who has OEC oversight of a weapon or information system that includes a human-software interface; his[1] mission performance, software, and MANPRINT analysts; and the Operational Tester. When the assessment involves user testing, the Operational Tester must follow the Guide for the preparation of data collection instruments, data collection, and preparation of descriptive statistical summaries and Usability Profiles.

1.2 **GETTING STARTED.** Use this Guide to evaluate or assess the usability and trainability of software interfaces. Although the Guide explicitly focuses on Information Mission Area (IMA) systems, it applies to any system that has a human-software interface. Since not all evaluators will have the same immediate goals, the first step in using this Guide is to decide where you need to start the assessment process for your system. If you have an upcoming opportunity (within six months or less), to collect data on software interface usability and trainability, go to Table 1.1 and review the tasks you must do first.

Table 1.1 A Quick Look for Those in a Hurry

ACTIVITY	COVERED IN:
1. Consult your MANPRINT Analyst for a rapid orientation	
2. Read or Review this Guide (scan App C-G)	Guide
3. <u>Gather or Review the latest</u>: Requirements Documents/Functional Description Critical Operational Issues & Criteria (COIC) Test & Evaluation Master Plan (TEMP) Critical Task List or Task Inventory System Manuals and Contractor Training Materials	
4. <u>Write or Revise</u>: Additional Operational Issues & Criteria (or Test & Evaluation Plan (TEP))	App A
5. <u>Select or Revise</u>: Data collection techniques	Ch 2 & App B
6. <u>Write or Revise</u>: Data Collection Sheets (If necessary, adapt App D-F directly into data collection sheets)	App C-F

[1] No terminology in the Guide is intended to be gender-specific.

Then see Table 1.2 for a quick overview of what you should do to prepare for the software usability assessment. If you have seven or more months before you can collect data on software interface usability and trainability, you will still have to do most of the tasks in Table 1.1. However, you may want to review the Guide before you consult your MANPRINT analyst. If you are reading this Guide just to see what it says, read all of it except Appendices B-G, which you can afford to scan.

1.3 BENEFITS OF THIS GUIDE FOR ASSESSING SOFTWARE USABILITY.
Here are the advantages you will realize when you use the methodology in this assessment guide.

- It is a valuable part of Continuous Evaluation. It covers all phases of system development and you can start using it in any phase.
- It provides quantitative measures of software usability that can show you whether efforts to improve software usability across the development process are succeeding.
- It is adaptable to any kind of human-software interface.
- It is forward-looking about developing robust data collection tools for operational testing.
- Early data collection instruments are based upon military standards for human-software interface design.
- Later data collection instruments are based upon inventories of user tasks (Critical Task Lists).
- You do not have to be a human factors expert to apply the principles and steps described in this Guide. (The methodology does require knowledge of the system being assessed.)

1.4 PURPOSE.
The purpose of this Guide is to help the Operational Evaluator and Operational Tester to:

- Develop plans to assess the usability and trainability of the software aspects of IMA systems during any phase of the development process.
- Define what the system and its software interfaces are and what they are supposed to do.
- Learn what types of data collection instruments are best.
- Develop appropriate checklists and questionnaires.
- Administer, collect, and analyze the data collected.
- Report the assessment results.

Table 1.2 Preparing for a Software Usability Assessment

PREPARING FOR A SOFTWARE USABILITY ASSESSMENT

1. Familiarize yourself with the system, its Critical Task Lists, its requirements documents, its T&E schedule, and any existing T&E planning documents.

   Find your next data collection opportunity. Identify the test proponent for the next test event. If necessary, get permission to observe the test event. Identify who will act as users during the next test event opportunity. Identify any restrictions on your access to Test Players for collection of OPTEC data.

   Get an inventory of the system's capabilities for each software interface. Learn as much as you can about the software capabilities that the system developers claim for each of the system interfaces. Try to get hands-on familiarization opportunities for your analysts, including hands-on user training.

   Get an inventory of the User Representative's job and task requirements for each software interface. If no formal task inventory exists, use the Critical Task List. Learn as much as you can about what people are supposed to be able to do using the system. This includes the functions, tasks, and interactions that are intended for each of the software interfaces you will assess. Consult the user requirements documents, such as the Functional Description.

2. Define the system's human-software interfaces and select those you will assess.

3. Write the Additional Operational Issues and Criteria (AOIC) or evaluation (or assessment) plan. For guidance, use Appendix A, Issues and Criteria.

4. Select data collection tools for each of the likely sources of data from Table 2.2. To develop data collection instruments, use the guidance in Chapter 2 and these Appendices:

B	Data Collection Tools
C	Items for Checklists and Questionnaires
D	Example Biographical Data Questionnaire
E	Example End-of-Task Questionnaire
F	Example Usability Profile Questionnaire

What is this Guide about?

How to assess the usability or User Friendliness of software packages that involve human interaction.
AND
How to measure the effectiveness and efficiency of human interactions with any software component.

1.4.1 What This Guide Covers. This Guide presents a comprehensive methodology you can use to assess the usability and trainability, or User Friendliness, of software packages that require human-computer interaction. The methodology does not dwell on the technical characteristics of software. It is performance-oriented. If the user cannot use a software program to do his job better in some way, then it does not matter how easy it is to interact with the software. So, this methodology is about measuring the effectiveness and efficiency of human interactions with any software package. These goals mean that the human-software interactions culminate in a good product, and do not require exorbitant resources of people, their time, their effort, or their skills. Although the primary emphasis of this methodology is upon software interfaces, it does not ignore the hardware and environmental characteristics that affect human performance.

1.4.2 <u>What This Guide Does **Not** Cover</u>. There are some aspects of software evaluation that this Guide does not encompass. Complete operational evaluations of software cover the operational effectiveness and suitability of software performance (besides software usability). Under Continuous Evaluation, they also measure the maturity of system software, primarily to ensure its readiness for test. These aspects of operational evaluation are beyond the scope of this Guide. Measuring specific characteristics or attributes of software programs, and measuring specific design features or operating characteristics of hardware components, are also outside the bounds of this Guide.

This Guide is <u>Not</u> About Measuring:

The "maturity" or "test readiness" of system software.

Specific characteristics of software programs.

Specific operating characteristics of hardware components.

Do not confuse this Guide with efforts to develop metrics for the software development process to find whether the software component of a system is ready for test. This Guide addresses the **usability of human-software interfaces**, emphasizing the capability of system software to allow optimum human and system performance.

1.5 **BACKGROUND.** This Guide is partly derived from previous research and development which resulted in the User Friendliness Handbook (UFHB).[1] That effort was undertaken for the Science and Engineering Directorate of OPTEC. Two versions of this Handbook have been produced and the methodology described in the

[1] OPTEC HANDBOOK. Handbook for the Evaluation of User Friendliness of Soldier-System Interfaces. May 1992.

Handbook has been used on several systems that contained embedded software. The information gathered in the preparation of the UFHB, insight gained during the writing and revisions to the UFHB, and lessons learned in the application of the methodology have all been drawn upon and incorporated into this Guide.

1.6 DEFINITIONS.

1.6.1 Human-Computer Interface (HCI). In the broadest sense, it is the medium through which a human being interacts with a computer, including all of its hardware and software. It is physical, in terms of the displays the computer uses to influence the performance of the human being, and the controls the human being uses to influence the performance of the other components of the system. It is **mental**, in terms of how the typical human being is likely to interpret system displays and to translate his own intentions into manipulation of the system's controls.

What is a Human-Computer Interface (HCI)?

It is the mechanisms through which the computer and the user communicate or interact. It includes everything seen on the screen, read in the documentation, and manipulated with a keyboard (or mouse, etc.).

1.6.2 Human-Software Interface. It is the part of the HCI that is software-programmable. More and more, the displays and controls of human-machine systems are becoming software-programmable. Witness the incorporation of displays and touch-sensitive controls into Video Display Terminals (VDT) in commercial aircraft, luxury cars, and military systems. It is the usability of software interfaces that this Guide addresses.

What is a Human-Software Interface?

It is the software-programmable part of the HCI -- the features of controls and displays through which the user interacts with the software components of the system.

1.6.3 User Friendliness of a Human-Software Interface. As an abstract idea, User Friendliness is a quality of system controls and displays that makes it easy for the human component of the system to interpret system displays and to manipulate system controls, to produce effective and efficient outcomes of human-computer interaction. As a dynamic idea that is important to the Operational Evaluator, User Friendliness is any measurable aspect of the usability or performance capability of <u>all</u> the interactions required at each human-software interface. To be **<u>effective</u>**, the product or outcome of a subtask done at a specific interface must be acceptable to the User Representative in terms of both quality and timeliness. To be **<u>efficient</u>**, the product or outcome of the interaction must be obtainable, both now and in the long haul, with a minimum of human resources. Human resources include the numbers, time, effort, and frustration of users expended to complete a task or interaction; and the skill levels, experience, training and education invested in the preparation of users to interact with the system. This Guide will show you multiple ways of measuring and assessing the effectiveness and efficiency of software interfaces.

What is the "User Friendliness" of a Human-Software Interface?

"User Friendliness" is a quality of system controls and displays that makes it easy for the human component of the system to **interpret system displays** and to **manipulate system controls**, to produce effective and efficient outcomes of human-computer interaction.

1.7 PRINCIPLES FOR THE MEASUREMENT OF SOFTWARE USABILITY.

Unlike physical measurements, software usability and trainability indices are indirect measurements. They are derived or inferred from measurements of the processes and products of human-software performance. They include measurements of the effectiveness (of products), and the efficiency (of processes), of human interaction with a computer-based system, using one or more human-software interfaces. Effectiveness is normally measured in terms of the **time** to produce the product and the **accuracy** of the product itself. Efficiency is normally measured in terms of the **level of effort** expended by the human component of the total system. This includes human-related resources, such as numbers of people or training resources required, as well as individual human effort. Appendix A is a set of generic Additional Operational Issues and Criteria (AOIC) that implements all software usability and trainability measurement principles mentioned in this paragraph.

1.7.1 <u>Real User Tasks are Essential</u>. User Friendliness can ultimately be assessed only in terms of the effectiveness and efficiency of the work that people do, using the system. It is the job and task requirements, not the system's capabilities

The usability of a software interface cannot be assessed independently of the tasks its users are required to do.

alone, that define the work that must be done. IMA systems that are otherwise User Friendly can still be deficient if they do not give the user the capability to do required work. Similarly, the User Friendliness of unneeded, unused system capabilities is entirely moot.

1.7.2 Multiple Indices are Required. It would be nice to describe software usability with a single index, but that is not realistic. A large-scale computer system usually has many software interfaces that are defined by the number of software components in the system and by the human roles associated with them. Each software component normally defines one software interface, even if it is part of an integrated software system. Each special-purpose software component, such as a personnel management system, might be associated with only one human role (i.e., a user, operator, or administrative role, such as mechanic, commander, personnel clerk, supply clerk, system administrator, or system security officer). A general-purpose software interface, such as a word processor, can be associated with many human roles. Some tasks may require a suite of software components to be used together consistently, and you might be tempted to define them as a single interface. This is most likely to apply to a cluster of special-purpose software tools (e.g., logistics tools). However, you should be reluctant to assess them as a single software interface. That is justified only if no person, in any of the roles that uses the software suite, has the slightest discretion whether to use one of the tools in the software suite to do the tasks required by his job.

Is this Guide going to give me a single "dipstick" to measure the User Friendliness of my system?

Sorry, that's a pipe-dream. You can't have a single usability index or profile unless your system has only one human-software interface, which isn't very likely.

1.7.3 Software Must be Effective to be User Friendly. It is not enough that users say they can make the software work. There must also be an effective outcome. For example, anecdotal data may show that a computer-assisted instruction (CAI) training program is easy to use. However, that is not sufficient evidence of User Friendliness, because the CAI program may not allow the user to practice and learn the behaviors he must do to get the most out of the software application itself.

1.7.4 Do Periodic Testing with Real Users. The best software usability assessment strategy for the Operational Evaluator is to collect data during periodic tests or demonstrations that have real users, to learn which human-software interactions are difficult or easy to do. As the program continues, the capabilities of the system are better defined and the inventory of required tasks becomes more comprehensive and detailed. The more you learn about the interactions that will take place at each interface, the more you can refine your data collection tools. The most common methods used to find the usability and trainability of software interfaces include: opinion questionnaires, tests of the user's level of knowledge before and after using the system, automated logging of user actions, observation of users working in their natural environment with their own tasks, or observations of users working on a set of representative standard tasks. The last two methods are especially suited for operational tests that allow users to do realistic work in an operational setting. Table 1.3 shows that the principles underlying this Guide have a solid basis in the software development community.

Table 1.3 Guidelines for Software Usability Testing

7 MAXIMS OF USABILITY FOR SOFTWARE DEVELOPERS[1]
1. Test the software, not the user.
2. Design for the user, not for the programmer.
3. Run the tests with real users.
4. Run tests early, and run them often.
5. Don't try to test everything at once.
6. Measure how well the software performs real-world tasks, not how well the built-in features work.
7. Always remember: Users will have problems that software designers never imagined.

[1] Reed, Sandy. "Who Defines Usability? You Do!," PC/Computing, Dec 92 (224-232).

CHAPTER 2
USABILITY EVALUATION METHODOLOGY

2.1 **METHODOLOGY OVERVIEW.** The methodology presented in this Guide consists of seven steps in which you investigate and assess the usability and trainability of the human-software interfaces of your system. Table 2.1 provides an overview of the steps necessary to evaluate human-software interfaces using the methodology described herein. By doing these steps at intervals during system development, you can focus and refine the data collection effort and estimate how well-suited the human-software interfaces are to its human users. If your assessment includes user testing, you will hand off part of Step 4, all of Step 5, and part of Step 6 to the Operational Tester.

2.2 **SELECTING DATA COLLECTION TECHNIQUES.** Table 2.2 (also found as Table B.1 and discussed in Appendix B), shows the possible types of software usability and trainability assessments you can do, given the type of test event and who controls data collection at the test. This table is very useful in long-range planning. It can help you assure that all software usability and trainability data collected by OPTEC during the system acquisition process, will be complementary. Further, the process will help you do early planning and negotiating with the Program Manager (PM) to get data you might not get if you ask for it at the last minute before a test.

PM/Contractor Data

You won't get any of their data if you don't ask for it. You still may not get it if you ask for it too late!

2.2.1 Selecting the Type of T&E Activity (Columns). To use the table, first select the type of test event from the columns in the table. The second column refers to technical test and evaluation activities (i.e., compliance testing). It also refers to contractor-conducted development or engineering testing. The third column refers to user testing activities, including Operational Test and Evaluation (OT&E). **You will select only one column that applies to a data collection opportunity.**

2.2.2 Selecting Who Controls Data Collection Tools (Rows). After you select the appropriate column, look at the rows that apply to your situation. Even when the primary purpose of the test is to allow the PM, technical tester, or system contractor to collect data, you can negotiate with the PM to have OPTEC personnel collect some data directly, and to have indirect access to data collected by PM/contractor personnel. Even in an OPTEC

Table 2.1 Summary of Steps to Assess and Report Software Usability

Summary of Steps to Assess and Report Software Usability

STEP 1: Familiarize yourself with the system.
- Find your next data collection opportunity. Scan the data collection methods and tools (see Table 2.2 or Table B.1).
- Get an inventory of the system's capabilities (as claimed by system developers and contractors), for each software interface.
  - System Tasks
  - System Complexity
  - Interaction Methods
  - Interaction Styles
- Get an inventory of the user representative's job and task requirements for each software interface.
  - User Task Requirements
  - User Experience
  - Rate of System Usage
  - Training

STEP 2: Define the human-software interfaces.
- Programs
- User Roles
- User Functions

STEP 3: Write the AOIC or evaluation plan.
- Issue, Scope, Criteria (MOPs), & Rationale
- Evaluation Approach
- Analysis of MOPs and Data Presentations

STEP 4: Develop data collection instruments. This will often be part of detailed test planning.
- Human-Software Interaction Categories:
  - Data Display
  - Data Entry
  - Error Management
  - Interactive Control
- Questionnaire/Checklist Item Selection
- Constructing Questionnaires

STEP 5: Collect checklist and questionnaire data.

STEP 6: Analyze data and report results.

STEP 7: Do the software usability assessment.
- Integrate Results from All Data Sources
- Discuss and Interpret the Results
- Answer the Software Usability Issues

NOTE: If the data collection event is a user test, the Operational Tester will do part of Step 4, all of Step 5, and part of Step 6.

test, there may be PM, technical tester, or contractor collected data that you would like to analyze (e.g., contractor "Help Desk" data). Therefore, you are not really choosing between rows. **You will normally look at both rows in a column to make sure you do not overlook a possible data source.**

Table 2.2 Data Recorders[1] and Data Collection Tools

Type of T&E Activity v. Control of Data Collection Tools

DATA COLLECTION TOOLS DEVELOPED & ADMINISTERED BY:	TECHNICAL TEST & EVALUATION ACTIVITIES (CD, ET, TT, DT, et al.)	USER TESTING & OPERATIONAL TEST & EVALUATION ACTIVITIES (EUT, LUT, IOT, FOT, et al.)
OPTEC PERSONNEL	1. Evaluator/Analyst[1] - Checklist/ Document Review - Checklist/ Planning Meetings - Checklist/ Observation - Checklist/ Hands on (?)[2] - Structured Interviews (?)[2 & 3] 2. Users (?)[2] - Questionnaires 3. Data Collectors/SMEs (?) - Checklist - Questionnaire - Performance Data[3]	1. Evaluator/Analyst[1] - Checklist/ Document Review - Checklist/ Planning Meetings - Checklist/ Observation - Checklist/ Hands on - Structured Interviews[3] 2. Users - Questionnaires 3. Data Collectors/SMEs - Checklist - Questionnaire - Performance Data[3] - SME Judgments 4. Instrumentation - Audit/Keystroke Trail - Video/Audio Data
PM, TT&E, OR CONTRACTOR PERSONNEL	1. Users (?) - Questionnaires 2. Data Collectors (?) - Checklist - Questionnaire - Performance Data - SME Judgments - Structured Interviews[3] 3. Instrumentation (?) - Audit/Keystroke Trail - Video/Audio Data	1. PM/Contractor Collected Information (?) - Checklists, Questionnaires, Structured Interviews, Instrumented data, Help Desk Logs, etc.

[1] Data recorders are the entities that initially record data. An evaluator/analyst is a data recorder when he writes comments and completes checklists. Data recorders in each cell are numbered.

[2] The question mark (?) after some data recorders and data collection tools indicates an area that you will have to negotiate with the PM (or Technical Tester/Evaluator), in order to collect or use this information.

[3] Interview guides for structured interviews should use the same data collection forms as for a questionnaire. The difference is that the evaluator/analyst (or a data collector), records the responses. (Some data recorded by data collectors may be short categorical responses (e.g., "Yes/No" or "Hard/Easy/Neither"), from Test Players. In this case, data collectors will read a short question (and its answers), from a performance data collection form and check the appropriate response block directly on the form.)

2.2.3 Selecting Potential Data Recorders. After you select the appropriate cells of the table, you will focus on the data recorders and the types of data collection tools to use. Depending upon many factors (e.g., budget, time schedule, support personnel), you will have to choose which options are best for you. However, note that the options listed within each cell are not mutually exclusive. You must always try to get as much as you can of both objective and subjective data on the same test events. You will get your best data on software usability from the performance data and the interview or questionnaire responses of Test Players who are typical of the users, operators, and administrators who will man the system after fielding. The benefit of real users as Test Players becomes increasingly greater to you if you can control or influence any of the following (which are not shown in any particular order):

- Test design
- Test schedule
- Development and administration of unit scenarios
- " " " " task scripts
- " " " " questionnaires
- " " " " interviews
- Data collectors
- Test directorate
- Test players
- Timing of test events
- Instrumentation
- Data management and reduction
- Data analysis
- Assessment reporting

All of the above probably add up to an OPTEC-conducted test. But, even when it is the PM's test, you can actively try to influence early test planning. If you bargain skillfully and early, you may get full or partial control over some aspect of testing or data collection that would otherwise <u>not have been included</u> in the PM's test.

2.3 STEPS TO ASSESS AND REPORT SOFTWARE USABILITY.

2.3.1 <u>STEP 1 - Familiarize Yourself with the System</u>. Before you begin an assessment of software usability and trainability, it is important to understand the system fully, including:

- What the system is intended to do
- System capabilities claimed by the developers
- User Representative's task requirements
- Where the system is in the development cycle
- The test schedule
- What kind of software interfaces the system uses
- Who will use, maintain, and support the system
- How best to find the strengths and weaknesses of the software interfaces

If you already have some data collection tools that were developed for earlier use in the program, you will refine those tools in this step, using the previous results and lessons learned.

2.3.1.1 Step 1a - Find your next Data Collection Opportunity. Review the system's test schedule to find the next event at which you can collect data. Do not overlook test events that are under the control of the contractor or the PM. Some of the information needed is:

- Where the users (or Test Players) will come from
- What kinds of tasks users will be doing
- Whether you can observe the action
- Whether you can sit at a terminal yourself
- Whether your analysts can attend user training courses
- Whether you can submit items for a user questionnaire (or even the entire questionnaire)
- Whether you can use your own data collectors to interview (or administer a questionnaire to) users

Once you have a feel for this type of information, review Table 2.2 (or Table B.1) to see what your data collection options might be. The information in this step will give you a preliminary idea of where you are in the usability assessment process and what options you have to collect the data you need to do the best possible assessment.

2.3.1.2 Step 1b - Get an Inventory of the System's Capabilities for each Software Interface. To assess human-software interfaces, you need to examine the system's capabilities (as claimed by the system developers), to support the functional proponent's missions, goals, and task requirements for the system. The following is a list of performance aspects to consider when examining the system in preparation for the creation of the questionnaire. Most of this information can be obtained from the system documentation such as Operational Requirements Documents, System Specifications, Functional Description, system manuals, or contractor training materials.

a. System Tasks. Consider what the system does and how. Some example tasks are: disseminating information, compiling information, computing targets, accessing a data base or providing situation updates.

b. System Complexity. Consider system complexity. System complexity in this context includes: what the system is required to do, the on-line documentation (help screens or explanations of the system), how the system is configured and operates, and the error handling capabilities of the system.

c. Interaction Methods. Consider how the system receives input and provides output. Some sample input methods are: keypad, keyboard, number pad, mouse, graphics tablet, trackball, lightpen, touch screen, or joystick. Output methods or characteristics include type and size of screen, graphics capabilities, and method of hard copy output.

d. Interaction Styles. Consider the interaction style(s) the system uses. Some example styles are: menus, form fill-in, command line entry, or icon manipulation/Graphical User Interface (GUI).

2.3.1.3 Step 1c - Get an Inventory of the User Representative's Job and Task Requirements for each Software Interface. The usability and trainability of human-software interfaces must be viewed in relation to what the functional proponent expects of the system. Therefore, you need to examine the functional proponent's missions, goals, and critical task requirements for the system. This need is explained in greater detail in Chapter 3 (para. 3.4.1) and Appendix C (para. C.4.1). The mechanism for developing extensive task inventories is to build upon the items in the **right column** of the four large tables in Appendix C. You will develop the task inventories by making master lists (one for each software interface), of all the tasks that users[1] will be required to perform using a software interface. The following is a list of things to do to prepare yourself to develop task inventories. Most of this information can be obtained from the User Requirements Documents, Critical Task Lists, and the System MANPRINT Management Plan (SMMP).

It is extremely important that you devote considerable **Continuous Evaluation** effort to expand and improve inventories of user tasks.

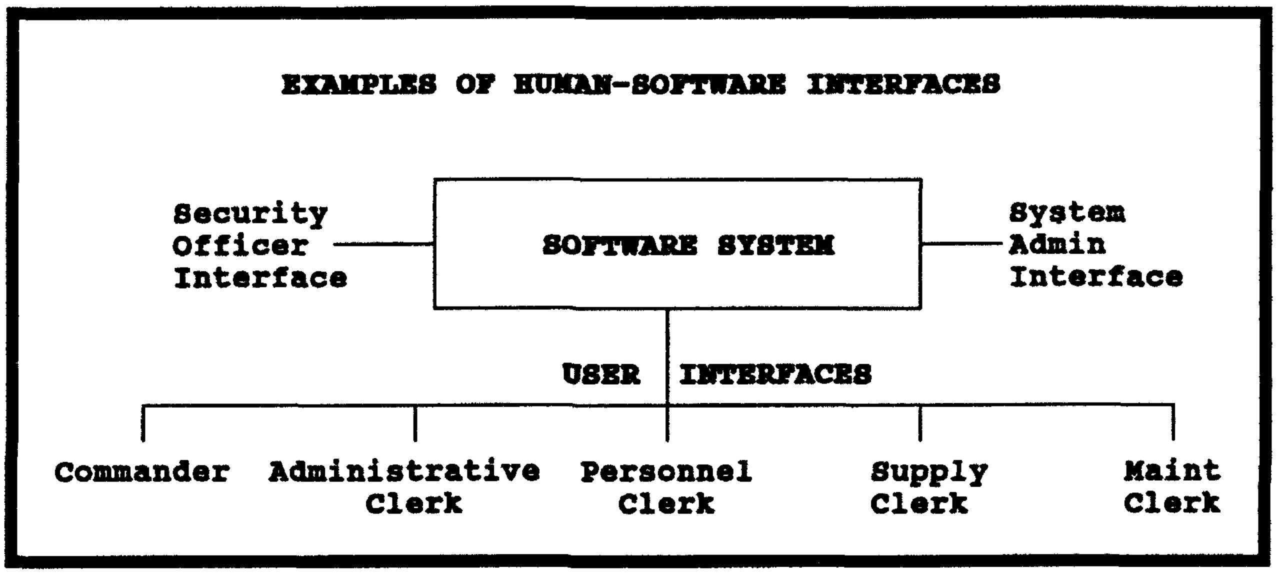
a. User Task Requirements. Consider what tasks the user will be assigned. Examples of tasks are: record keeping, file maintenance, communications, target tracking, data management, simulation, and status monitoring.

b. User Experience. Consider relevant characteristics of intended users, including experience, position, and duties. The more experience the user has with the system (and software systems or computers in general), the easier it is for the user to operate software intensive systems.

[1] This usage of "user" is intended throughout the Guide to refer to any person who physically interacts with the system. Users include not only those who use the system to grind out work for a living, but all those people who operate and support the system in administrative and maintenance roles. [In some cases "user" refers to the User Representative (i.e., the Functional Proponent), which will be obvious from the context in which it is used.]

c. Rate of System Usage. Consider system usage rates (i.e., Hourly, Daily, Weekly, or Rarely), by type of user. The more the system is used, the easier it will be to retain knowledge of the system and gain experience on it.

d. Training. Review training plans for the system and consider the type of training to be provided to the user. The types of training range from classroom-blackboard training to on-the-job-training. Refresher courses may be offered to users periodically, or to users who have not used the system for some time.



2.3.2 STEP 2 - Define the Human-Software Interfaces. Table 2.3 shows the steps you will take to define the human-software interfaces for the system and to select those you will assess. The number of human-software interfaces in a computer system is the result of a three-way alignment of the following factors:

Programs: Software applications or programs on the system (categorized by General-purpose, Special-purpose, and Customized)

User Roles: Roles of people who interact with the system (categorized by Administrators, Operators, Dedicated Users, and Occasional Users)

User Functions: Functions done by people who interact with the system (categorized by Data Entry, Data Access, Data Processing, Data Reporting, System Operations, and Combinations)

2.3.2.1 Alignment of Programs, User Roles, and User Functions. For a very simple system, these factors could all be perfectly aligned so any of t..e three factors would define the same set of interfaces. This might happen if all the programs on the system are special-purpose or customized and if no user roles share the same tasks. However, in the nightmare extreme, each combination of program, user role, and user function would define a unique human-software interface. Fortunately, the nightmare case is highly unlikely -- it means that any function or task could be performed by any user role using any software program!

Table 2.3 Defining Human-Software Interfaces

DEFINING HUMAN-SOFTWARE INTERFACES

1. Identify the various software programs and classify them as follows:

   - General Purpose
   - Special Purpose
   - Customized

2. Identify the human roles that will interact with the system and classify them as follows:

   - Operators
   - Administrators
   - Dedicated Users
   - Occasional Users

3. Identify the types of user functions that will be done at each interface and classify them as follows:

   - Data Entry
   - Data Access
   - Data Processing
   - Data Reporting
   - System Operations
   - Combinations

4. List the intersections of the classifications of each of the factors above. This is a comprehensive, ideal list.

5. Collapse similar intersections from the ideal list to make a manageable list of software interfaces for testing.

6. Make adjustments as necessary throughout the planning stages.

2.3.2.2 Alignment of Software Interface Factors in Complex Systems. Complex systems will seldom have a perfect alignment of programs, user roles, and tasks. Therefore, selecting a manageable set of human-software interfaces for a specific operational assessment or evaluation is always a compromise. First, wanting to have a pure, simple analysis of each software interface will force you to define as many interfaces as you need. Second, wanting to keep the data collection and reduction processes manageable will force you to define fewer interfaces than you might like. Though it may not be possible, you should define interfaces so each Test Player will have to answer a Usability Profile questionnaire about only one software interface. Third, the number of human-software interfaces you can test is affected by the availability of test resources such as time, money, and numbers of Test Players, data collectors, and hardware installations. As a minimum, you should examine at least one interface for each user role that interacts with the system. But be prepared to be flexible during OT&E planning.

How Many Software Interfaces are There?

As a minimum, define at least one interface per user role.

2.3.2.3 <u>Coordination of User Task Scripts with Defined Software Interfaces</u>. It is important to ensure that the tasks developed for real-user testing are compatible with how you have defined the interfaces. For example, if you pool all general-purpose programs into one office automation (OA) interface, you may find that one role never uses most of the software tools in that interface. Therefore, analysis considerations may force you to define a second interface for the subset of the office automation applications that <u>are</u> used by that role. In another case, you may find that some tasks being scripted for use in a test cannot be completed without using two different software programs. The Test Player has no choice but to use them both. If the tasks are realistic and common, you might define an additional interface in terms of that role, the set of functions, and the two software programs. However, if the use of two software programs on a single task is possible, but not necessary, you might divide the script for the larger task into two scripts, one for each software program. This will avoid having one test event involve the use of two software interfaces.

2.3.3 <u>STEP 3 - Write the AOIC or Evaluation Plan</u>. Once the preliminary information has been gathered, reviewed, and digested, you need to create an evaluation plan. You will probably do this as part of Test and Evaluation Plan (TEP) development. An evaluation plan will allow you to record your intended issues, scope, criteria, rationale, evaluation approach, measures of performance (MOPs), data required, data collection, analysis, and reporting procedures. Appendix A provides a starting point for developing the usability and trainability portion of Chapter 2 of a TEP. You can customize Appendix A to your specific system and place it into the appropriate sections of the TEP. See the MANPRINT analyst in your directorate or in the OEC Technical Integration Division for an electronic copy (WordPerfect 5.1 format) of the main body of Appendix A.

2.3.4 STEP 4 - Develop Data Collection Instruments. In this step, you will select appropriate checklist and questionnaire items and tailor them to the particular system you are assessing. Use Appendix C as a starting point. If the data collection opportunity is a user test, the Operational Tester will do most of Step 4. However, we recommend that you or your MANPRINT analyst do the selection of checklist or questionnaire items yourself, then hand them off to the Operational Tester.

2.3.4.1 Human-Software Interaction Categories. The checklist and questionnaire items in Appendix C are divided into four categories of human-software interactions that apply to every software interface. In Appendix C, you will find a separate large table of checklist items and questionnaire items for each interaction category. The following paragraphs describe the four categories.

Categories of Human-Software Interactions

Data Display

Data Entry

Error Management

Interactive Control

a. Data Display. This category of human-software interactions encompasses the **methods and data formats through which the system conveys information to the user.** Most software systems employ some type of screen display such as Cathode Ray Tubes (CRT), Large Screen Optical Projection, Dot Matrix/Segmented Displays, or Light Emitting Diodes (LEDs). This category examines: the organization, structure, coding, labeling and format of information presentation on the screen display. Subcategories within Data Display are: Display Format, Display Content, Display Coding, Dynamic Displays, Display Suppression, Tabular Displays, Graphical Displays, and Audio Displays.

b. Data Entry. This category of human-software interactions encompasses the **methods and data formats through which the user conveys information to the system.** These methods include: keyboards, keypads, joysticks, lightpens, touch screens, trackballs, mice, grid pads, optical character recognition (scanners or page readers), and voice input. This category examines: the organization, structure, methods, editing and format of information input by these methods. Subcategories within Data Entry are: General, Position Designation [Cursor], Data Forms, Other Data Processing, Hardware Control Methods, and Text/Program Editing.

c. Error Management. This category of human-software interactions encompasses the **methods by which the software interface provides user help, security, data protection, and system protection.** This category examines: recovery from errors, avoidance of errors, protection of data, security of data, documentation (such as help screens), system records, system aids, and feedback. Subcategories within Error Management are: General, Multiuser Systems, User Identification, Data Access, Data Entry/Change, Loss Prevention, Normal Operating Feedback, Error Feedback, Prompts, Defaults, Job Aids, and Usage Records.

d. Interactive Control. This category of human-software interactions encompasses the **methods by which the software interface allows the user to manipulate, direct, and govern system operation.** The dialog type, organization of transactions, varieties of manipulations, system responsiveness and the ability of the user to control the system are examined in this category. Subcategories within Interactive Control are: Dialog Type (seven types), Transaction Selection, Interrupt, and Control Relationship.

Table 2.4 Preparing a Software Usability Checklist

PREPARING A SOFTWARE USABILITY CHECKLIST	
1.	Develop one checklist for each interface that was selected.
2.	Go to the four long tables in Appendix C. Look at the left column under each of the four categories of interactions.
3.	Select all subcategories that apply to the particular interface selected.
4.	Select all checklist items that apply to the particular interface selected.
5.	Write additional checklist items from your current knowledge of the system and its intended functions, tasks, and interactions. The goal is to develop a comprehensive set of checklist items that addresses all expected human-software interactions that will be required (and possible) at the interface.
6.	Use the entire list of checklist items that you have selected. If that is infeasible, make certain you include each checklist item that covers a critical task. Then use a table of random numbers to select from the pool of remaining items. You should have a total of 15-25 items from each of the four categories.
7.	Score each interface in each of the four categories of interactions by tallying the percentage of responses that show a favorable response to the usability and trainability of the interface.

2.3.4.2 Checklist/Questionnaire Item Selection. Table 2.4 summarizes the steps you will take to develop a software usability checklist. To determine which checklist/questionnaire items listed in Appendix C apply to a particular system, you must select or delete items using what you learned while examining the system's capabilities and the user's task requirements. Review the four large tables in Appendix C to find those items you will use and those items you will discard. The process works as follows: If the description listed below the subcategory applies to the software interface you want to assess, then consider the items under that subcategory for use in the checklist or questionnaire. If you do not know what the description means, scan the items under that subcategory. If the description does not apply, then those items do not need to be included. After you have selected applicable subcategories in Appendix C, review each question or item individually to see whether it applies to the software interface you are evaluating. Here are some points to remember as you build upon the items in Appendix C:

a. Construct a unique set of items for each software interface you will assess.

b. The maximum number of items used in a single data collection form should not exceed 100, especially for a questionnaire. Include or omit items at your discretion. Notice that the lists of items in Appendix C are not exhaustive. No document could contain all possible questions for all systems. The purpose of this method is to use Appendix C as a solid starting point from which you can develop useful data collection tools. Other items can, and should, be added as necessary.

c. Not all items and questions are applicable to all respondents in all circumstances. You can partially eliminate this problem if you define software interfaces carefully. The evaluator must consider what knowledge the chosen respondent has about the system and ask only those questions that the respondent can answer. For instance, users should not be asked detailed questions about software design or system processing time on a questionnaire administered after a system test. They would probably not retain that information very long, so capture perishable memories immediately after a Test Player completes a task.

d. Many checklist and questionnaire items will have to be reworded to customize them to the system being evaluated.

2.3.4.3 Constructing Questionnaires. Table 2.5 summarizes the steps you will take to develop a software usability questionnaire. Once you have selected the items for the checklist or questionnaire, organize them and place them in the correct format to administer to users. The users may be formal Test Players or any people you can collect data from who are very familiar with the software interface. If there are no "representative users" in a data collection event, you might still administer questionnaires to members of the acquisition team (e.g., designers, analysts, programmers, or independent evaluators). Of course, if the data collection opportunity is some type of user test, you will hand off the lists of checklist and questionnaire items to the Operational Tester who will prepare the data collection instruments. Appendix C gives specific instructions on how to construct software usability checklists and questionnaires. Appendices D-F are examples of questionnaires that can be administered in groups. For convenience, when the word "questionnaire" is used alone in the remainder of this chapter, it represents either type of question format chosen by the evaluator (i.e., checklist or questionnaire item), including the use of questionnaire forms as guides for structured interviews. Appendices A and C-F exist in electronic form and are available from the MANPRINT analyst in your directorate or in the OEC Technical Integration Division.

Table 2.5 Preparing a Software Usability Questionnaire

PREPARING A SOFTWARE USABILITY QUESTIONNAIRE

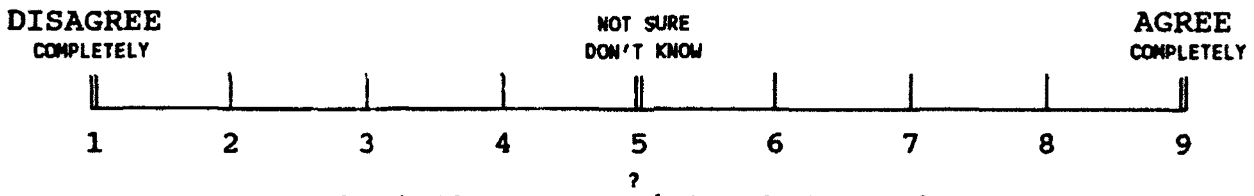
1. Develop one questionnaire for each interface that was selected.
2. Go to the four long tables in Appendix C. Look at the <u>right column</u> under each of the four categories of interactions.
3. Select all subcategories that apply to the interface.
4. Select all potential questionnaire items that apply to the interface.
5. Write additional questionnaire items from your current knowledge of the system and its intended functions, tasks, and interactions.

   The goal is to develop a comprehensive set of questionnaire items that addresses all expected human-software interactions that will be required (and possible) at the interface.
6. Circulate the list of interactions to several SMEs for help in finalizing the master list of items.
7. Rewrite potential questionnaire items as necessary.
8. Use the entire list of questionnaire items in the Post-Test Usability Questionnaire for each software interface.

   If that is infeasible, make certain you include each questionnaire item that covers a critical task. Then use a table of random numbers to select from the pool of remaining items. You should have a total of 15-25 items from each of the four categories.
9. Score each interface in each of the four categories of interactions by calculating the mean responses on a dimension of favorableness toward the usability and trainability of the interface.

2.3.4.3.1 <u>Biographical Data Questionnaires</u>. When you collect software usability data from real users, include biographical data questions about the experience and training of the respondents. These questions can be selected from Appendix D. The answers to these questions can help you diagnose anomalies in the responses to the interface questions. They can also help you find out how easy the system must be in order for the respondents to use it.

2.3.4.3.2 Numerical Response Scale Items. When writing human-software interface items, use predominantly a numerically-scaled response format. All scaled response items on a questionnaire should use a nine point scale. Question stems and scales should be worded so the highest number is always the most favorable toward the usability of the software interface. The response dimension you use for all scaled response items should be defined by bipolar pairs of adjectives, verbs, or adverbs, as shown below. It is best to use a single response dimension definition throughout the whole questionnaire. You can normally word all question stems about human-software interactions so they are compatible with the Agreement-Disagreement dimension used below. Such question stems are statements about the ease (or lack of difficulty) of using system controls or of understanding system displays. **All "Questionnaire Items" in Appendix C are in this format.**



(Select the appropriate whole number)

2.3.4.3.3 An Alternative to Narrative Comments. Comment data can often provide the most insight into the usability and trainability of a software interface, but it can also be the most difficult to reduce, analyze, and report. In place of open-ended questions that produce free-format narrative responses, you may choose to use scaled response items, followed by a space for the respondent to write brief comments. This can be very useful when there are many respondents, but few analysts.

2.3.4.3.4 Tailoring Items and Questionnaires. The sequence of the questions or items should not confuse the respondents. Pose the most general questions first, then become progressively more specific. Cluster related items so there are no abrupt changes from topic A to topic B, then back to topic A. The questions or items should be understandable, should yield useful data or information, and should not unduly influence the respondent to answer a certain way. The items in Appendix C are written at or below the 11th grade reading level. If you think a question or item in your list could be misunderstood, make it easier to understand without changing the intent of the question. Also, reword all questions or items to reflect system-specific usage, notation, and terminology as found in the system documentation or as taught during Test Player training. For instance, different systems often use different terms to mean the same thing:

- The device used to indicate the current position on the screen may be called a "pointer", "cursor", or "indicator".
- Tasks on the system may be referred to as "transactions" or "data calls."

2.3.4.3.5 Field Testing the Data Collection Forms. You will rarely have a chance to do formal field testing of the data collection forms with people drawn from the same pool as Test Players. Fortunately, you will be using the entire Continuous Evaluation process to refine them anyway. However, if possible, you will have several people who are familiar with the system (and with the test), complete each of the questionnaires. Check the results to see if they are what you expected. Revise the items on the data collection forms as necessary. Repeat this several times. As a minimum, you should have several colleagues on the evaluation team or in the test directorate do a rigorous review of the data collection forms before they are used. This must be done early enough that the finalized data collection forms are available for data collector training.

2.3.5 STEP 5 - Collect Checklist and Questionnaire Data. The next step is to finalize and administer the questionnaires. The Operational Tester will normally do this step for a user test. When administering the questionnaires, data collectors (and test directorate personnel, observers, evaluators, and analysts), **must** be careful not to influence the respondents' answers in any way. This means that none of the opinions about the usability of the system, by any member of the acquisition team, should ever be expressed within earshot of a potential respondent.

> **WARNING:** No member of the acquisition team should ever express his or her own opinions about the usability of the system within earshot of a potential Test Player or questionnaire respondent.

2.3.5.1 Collecting the Data. Checklists furnished to members of the acquisition team (e.g., designers, analysts, programmers, or independent evaluators), should be completed by answering YES or NO relative to existing design, code, screens, programs, documentation, and prototypes. Questionnaires completed by respondents reflect their opinions of the usability and trainability of the software interface after training, testing, or trial use of the system. The questionnaire administrator should ensure that any necessary support facilities and equipment are available and are used properly (e.g., a room for administering the questionnaire, or video or audio equipment to record interview sessions or operational use of the system). He should also ensure that respondents answer <u>every</u> question completely, documenting all discrepancies if possible. In an operational test, the respondent should perform tasks with the system in accordance with standard Army procedures, technical manuals, and operating instructions.

2.3.6 <u>STEP 6 - Analyze Data and Report Results</u>. Information on the analysis of software interface usability data is in Appendix A, Issues and Criteria. Appendix A is also the starting point for the assessment report. Appendix G, Checklist and Questionnaire Data Displays, shows how data from the different types of checklists and questionnaires can be reduced and displayed as meaningful results. These steps include how to construct the Usability Profile. If the Operational Tester has collected the data for you, the Usability Profile should be included in the Test Report.

2.3.7 <u>STEP 7 - Do the Software Usability Assessment</u>. The blueprint for this step is the evaluation plan or the AOIC you have written. If you have followed Appendix A fairly closely, this part will not be difficult. If you have not done so, and have no TEP or AOIC to work from, start with an electronic copy of Appendix A and use it as your draft for the software usability portion of the assessment or evaluation report. Appendix A contains guidance on all the statistical analyses you might want to use for all the software usability data. It also contains guidance for the reduction of the performance and other data you need.

Appendix A provides a starting point for your software usability evaluation plan. Tailor it as necessary when you write the evaluation plan. It will also provide a head start on the evaluation report.

2.3.7.1 <u>Integrate Results from All Data Sources</u>. This is a process of collating data according to the questions you want the assessment to answer. These questions are the issues and subissues in your copy of Appendix A or in the draft assessment report. You may want to refer to Table 2.2 (or Table B.1),to refresh your memory about all the possible sources for data on software usability.

2.3.7.1.1 <u>Checklists and Questionnaires</u>. Step 6 covers showing the results of your own checklists, questionnaires, and structured interviews. Table 2.2 shows that some checklist and questionnaire data may be available from the PM, Technical Tester, or contractor. It is difficult to predict the format of this information. If the format is similar to your own, you may be tempted to pool it with your own data. However, you must be

If You Get Data that OPTEC Did not Collect

Don't pool other people's data with yours. Use it as background information as you interpret your own. Use it to help you put your own results into words. Or, analyze it separately.

very careful about doing this, because other people's data may not have been collected under conditions comparable to your own, nor collected with the same rigor. You may never know what the vital element is that makes it unwise to pool your data with that of others. Therefore, always use other people's data as background information to help you interpret your own data. This can be helpful when you get unexpected or anomalous results and have to do some diagnostic work after you no longer have access to Test Players. If you do want to reduce, display, and analyze someone else's data, you will probably want to keep it separate from your own.

2.3.7.1.2 Performance Data. If data reduction plans were coordinated with you in pretest planning, the performance data will already be in a familiar format as Task Completion Rates, Task Success Rates, and Task Execution Times. Display the descriptive and inferential statistical summaries of the performance data in the appropriate place in your draft assessment report (or copy of Appendix A).

2.3.7.1.3 Instrumented Data (System Audit). If you have helped the instrumentation planners define the formats for reduced audit data, it should also be in a familiar format, related to the manually collected performance data. However, Appendix B (para. B.6.1) explains that this seldom happens. If the data format is not customized to your assessment requirements, and you have a pressing need for information that you know was captured, you will have to develop your own data reduction schemes from displays of the instrumented data. Uses will be to augment manually collected performance data and to explain inconsistencies and anomalies in other data.

2.3.7.1.4 Instrumented Data (Audio/Video Tape). You can use this type of instrumented data in several ways:

- As background information to help you interpret other data
- As a way to revisit the test site for diagnostic work to explain inconsistencies in other data
- As the basis for observational checklists when real-time observation is not possible
- For systematic coding of the interaction processes, including real-time comments by Test Players
- To augment manually collected performance data

2.3.7.2 Discuss and Interpret the Results. The goal is to explain why and how the data lead you to the answers you give to the issues, so readers of the report will draw the same conclusions. The reported facts, logic, and the draft report (or Appendix A), should be your guide. The lists you provide of the strengths and weaknesses of the software interfaces are especially valuable. The system developer can use them to improve the software interfaces to increase the probability of mission success.

2.3.7.2.1 About Interpreting Results. In Appendix A, Issues and Criteria, each time a Measure of Performance (MOP) is introduced, it is followed by a notation in brackets. The notation shows which end of the measurement scale normally indicates greater software usability. That is very useful information, but it must be used carefully, because **numbers never stand alone**. You must be completely aware of the circumstances under which the data were collected before you can use it to make a particular finding. In fact, the test director must ensure that data collectors, and all test directorate members, understand how valuable their own information is about how a Detailed Test Plan (DTP) was actually implemented. There are always unexpected events that can make the numbers mean something different from what they seem. The problem is that the test directorate will not have a structured data collection form to cover every conceivable factor that could bias the data. Here are some examples of what can happen:

> **Numbers Never Stand Alone**
>
> You can never have too much background information on the circumstances under which the data are collected.

- Shorter task completion times normally mean that one type of task is easier to do on the system than another type of task.
    - But, some applications (e.g., graphics packages), can be so fascinating to users that they waste a lot of time playing with the system. However, they probably would not spend much time playing with an application that is user hostile.
    - But, one type of task may be done predominantly under the supervision of a few data collectors who incorrectly believe they are allowed to offer tips and suggestions to users having trouble.
- Frequent rest-breaks during a task can mean that the application is so user hostile that the user takes rest-breaks to avoid working on the task.

- •• But, a user may have stayed up late the night before, and may really need to frequent the coffee machine.
- •• But, certain health problems or physical conditions can make a user take frequent breaks.

The point is to emphasize that: **You can never have too much background information on the circumstances under which the data are collected.** This is why it is good to have data on a single test trial collected by multiple data recorders. It is also why the test directorate must encourage all its people to make note of things that happen that are not covered in the DTP.

2.3.7.3 Answer the Software Usability Issues. Answer the issues and subissues in your draft assessment report (or copy of Appendix A). Make your answers as direct and complete as possible, including any qualifications you must make. However, avoid putting the rationale or explanation for any qualifications in your response statement. If something still needs to be explained, put it in the discussion of results. Do not be unnecessarily blinded by your results as you write your answer. You need to know them so you can answer the question, but you must focus on the question itself to provide a clear answer.

RTP

Read The Problem!

Remember this as you write your answer to an issue.

2.4 **ABOUT SOFTWARE USABILITY PROFILES.** These will remain the core of the operational assessment of software interface usability, especially when it comes to making comparisons across interfaces, time, and systems. There are several reasons for this:

- Software interface usability data must ultimately come from hands-on users, as objective performance data or self-report opinion data about real human-software interactions
- Post-test questionnaires may be the only kind of data you can normally expect to get across the entire spectrum of technical and operational testing
- Usability Profiles are derived from non-disruptive post-test questionnaires
- Usability Profiles are based upon the selection of human-software interaction items from an exhaustive pool of items that apply to each software interface

WHAT IS A USABILITY PROFILE?

Test Players' numerical ratings of the usability of a human-software interface generate usability scores for four categories of human-computer interactions:

Data Display	Error Management
Data Entry	Interactive Control

Each score is a mean rating of 15-25 items that describe the ease-of-use of the software interface to do required tasks. Scores are on a scale of 1-9, where "1" is unfavorable, "5" is the theoretical neutral point, and "9" is favorable. The four scores, plus the aggregated score across all four categories, make up the Usability Profile for that human-software interface.

Appendix G contains examples of a Usability Profile in Table G.5 and Figure G.1.

CHAPTER 3
REPETITIVE ASSESSMENTS

3.1 **CONTINUOUS EVALUATION.** To do a Continuous Evaluation of the suitability of a system's human-software interfaces, you should try to do real-user assessments at several points before reaching the window for operational testing. By repeating real-user assessments at successive intervals, you can track Usability Profiles of the system interfaces across the software development process. For simplicity, you may want to run several series of real-user assessments that look at only a few software interfaces at a time. Early real-user assessments are not always possible. However, information drawn from them and from checklist assessments can Le combined with other data accumulated through the Continuous Evaluation effort. The data will help you estimate whether software usability is improving during the development process.

This methodology can give valuable information about the potential usability of a software system, whatever its stage of development.

3.2 **PREDICTING SOFTWARE USABILITY.** The results of repetitive real-user assessments of software usability can be called Longitudinal Usability Profiles. These compare successive Usability Profiles to those of earlier assessments. They allow you to measure the improvement of software usability over time. Since multiple assessments establish a trend line, they improve your ability to predict future software usability results. Once this methodology has been applied to several systems, it will even be possible to make comparisons with other systems. As OEC increases its experience with the trend data of Longitudinal Usability Profiles, Operational Evaluators will be able to anticipate the software usability results of user tests with greater confidence.

3.3 **COMPARING USABILITY PROFILES ACROSS ASSESSMENTS.** There are several ways to compare Usability Profiles from one assessment to another.

3.3.1 Longitudinal Comparisons (Same System). This type of assessment compares Usability Profiles of the same software interfaces across successive assessments during the system development process. This is the type of comparison that will allow you (or your replacement) to estimate, under Continuous Evaluation, the progress that system developers are making in the area of software interface usability.

3.3.2 Single Time-Slice Comparisons (Same System). This type of assessment compares Usability Profiles across the different software interfaces of the system, as measured in the same data collection event. You can use these comparisons to estimate which software interfaces and software components are in the greatest need of attention by software developers. These observations can be a very important part of Continuous Evaluation.

3.3.3 Single Time-Slice Comparisons (Different Systems). This type of assessment includes direct comparisons between two or more systems. Direct comparisons may not be allowed under certain circumstances of competition sensitivity (e.g., in a Competitive Demonstration). However, extensive use of the Usability Profiles will eventually make historical comparisons very useful to operational evaluation. Where the data exist, you can compare systems in development with their predecessor systems and with similar fielded systems.

3.4 EARLY PREPARATION FOR QUALITY OPERATIONAL TESTING.

One benefit of doing repetitive software usability assessments is the chance to revise and improve your assessment methodologies, issues and criteria, data collection tools, and data analysis tools as the system progresses through development. Lessons learned from previous assessments can only improve successive revisions of the above, helping to enhance the quality of later operational testing.

Early Preparation for Quality Operational Testing

Use successive software usability assessments to:

Refine Additional Operational Issues And Criteria.

Refine experimental designs for operational testing.

Refine analysis tools.

Refine data collection tools

3.4.1 Inventories of User Tasks. An important idea behind the measurement of Usability Profiles is that what must ultimately be measured is: The ability of individuals and units to use the system to do the work they must do to accomplish their missions. You must learn whether the software interfaces will enhance individual and unit productivity on the jobs and tasks that the functional proponent requires. It is not good enough merely to test the usability of the software capabilities of the system, some of which may not contribute at all to the productivity of most system users. You must eventually have exhaustive, detailed task inventories for each software interface, to the level of task elements and below. The format of these task inventories is simple. You will build upon the lists of Questionnaire Items found in the right-hand columns of the tables in Appendix C. Your task inventories will be written about much smaller units of behavior than the Critical Task List or task inventory normally used for training development purposes. They will probably be a mix of both system capabilities and task requirements. It will take repetitive assessments of the system to develop exhaustive lists of task items, tailored to the system and organized by software interface. It will also take emerging knowledge of refinements to the system and to user requirements to build adequate lists of task items.

You must eventually have detailed task inventories for each software interface.

The format of these task inventories is simple.

Build upon the lists of items in Appendix C.

3.4.2 Data Collection Instruments. One value of repetitive assessments of a given software interface is to improve the data collection instruments. If an assessment is begun early in the development of a software system, you will probably have to begin by developing software usability checklists. For this, you will use this Guide and what you then know about **what the total system** (of people and machines), **is supposed to do**. Much of the latter knowledge will come from the functional proponent's broad requirements. Appendix C gives you an excellent place to start developing your checklists. You will revise the checklists you use after each assessment, and as user requirements and system capabilities change. In Appendix C all Checklist Items have corresponding Questionnaire Items. As you build customized checklists, you should also start to build the customized lists of Questionnaire Items that you will need. As the system matures, and real users begin to interact with it, you can develop and refine your lists of Questionnaire Items. You can delete, write, and rewrite Questionnaire Items for each software interface you have defined. After each real-user assessment, you will further refine these lists of Questionnaire Items.

Revise and update your lists of Questionnaire Items after successive software usability assessments.

3.4.2.1 Avoiding Very Long Questionnaires. Some software interfaces may be very simple, and the inventory of user tasks so short, that you can use the entire list of potential items in the Post-Test Questionnaire. However, for some complex software interfaces, there will be so many task items in a complete list that you cannot use them all. To reduce the burden of an endless questionnaire (to Test Players and data analysts alike), you will first select all the task items that cover critical tasks. To round out the rest of the questionnaire, you should randomly select from the remaining task items to ensure that your assessment uses a set of task items that represents the entire list. This unbiased sampling of the remaining task items is advisable when the selection pool is very large and you want to ensure that each questionnaire is a valid representation of all the items in the pool. The next time you assess the same interface, we recommend that you reconstruct the questionnaire as above, starting with the updated list of potential task items. This is especially important as software changes are made, tasks are added or deleted, and different tasks become critical.

Revising Questionnaires

The continuity between successive uses of questionnaires is based, not upon merely revising the old questionnaires, but in refining the pools of task items from which the Questionnaire Items are drawn.

3.5 **USING DATA COLLECTION INSTRUMENTS ACROSS SYSTEMS.** The same questionnaire or checklist should normally not be used for different systems. Each system is unique and requires specifically tailored instruments to collect the necessary information. Toward this end, you should not simply copy the example questionnaires shown in Appendices D, E, and F without customizing them for the system under test.

APPENDIX A
ISSUES AND CRITERIA

This appendix contains draft MANPRINT Additional Operational Issues and Criteria (AOIC) about the usability of a computer-based system's human-software interfaces.[1] Use the electronic copy of the main body of this appendix as a first draft for appropriate portions of Chapter 2 of the combined OEC/TEXCOM Test and Evaluation Plan (TEP), or for any earlier document that must contain AOICs. Optimum conditions to address these AOICs occur during OPTEC-controlled user testing.

Defining Major Software Interfaces

1. A major software interface is a medium through which the functions of a human role interact with the functions of a software program. There is normally one software interface for each major program or application, even if the programs on a system are integrated. However, the need for simplicity in testing and data collection may dictate that similar combinations of software programs(s) and human role(s) and function(s) define a single software interface.

2. Most IMA systems will have software interfaces for each general-purpose user application, for system administrators, for operators, and for security officers. Other common interfaces may be for the management of logistics, supply, maintenance, or personnel, and for command & control.

3. If a software program is adapted for use with two distinctly different human-hardware interfaces or software operating systems, it should be treated as two software interfaces.

2.3.x **Issue X** How <u>suitable</u> are the system's human-software interfaces for their intended users?

2.3.x.1 **Subissue X.1** How <u>effective</u> are human interactions with the system's software interfaces?

---

[1] This appendix specifically addresses IMA systems because they are potentially the most complex cases for the analysis and evaluation of human-software interfaces. However, this appendix can be adapted to any computer-based component or system that has a human-computer interface.

2.3.x.1.1 **Scope.** This issue addresses the task effectiveness of human interactions at each of the system's major software interfaces. Task effectiveness is assessed in an operational setting with test players drawn from user populations, working on realistic tasks. Special-purpose or customized software interfaces require test players who presently work in the jobs that will use those interfaces. General-purpose software interfaces may use test players drawn from a variety of pertinent jobs.

a. The units of analysis for test data (i.e., the types of test events or trials), are carefully defined during detailed test planning. For individual tasks, including sequential chains of tasks by two or more people, the unit of analysis is an attempt by one soldier or government employee to do a (scripted) task assigned by the tester. For complex collective tasks, the unit of analysis applies to the work of a team of interdependent test players. Test players are interdependent if they must work together, interactively and simultaneously, to finish a task. The focus of data collection and analysis in user testing is to capture and analyze all objective and subjective data that are pertinent to each test trial, including the data needed to address this issue. Checklists for compliance with minimum user requirements are applied to the outcome or product of each task trial by analysts or expert judges.

b. A task trial must be defined carefully so measurements will be made only of the test player's direct contributions to system performance. First, a Task Execution Time should include only the time needed for the user to interact with the system to set up and initiate a process. It must not include any time that the system takes to process data, print, back up the hard drive, etc., while the test player watches with folded arms. Second, when analysts or expert judges rate the success of a task trial, they must restrict themselves to those outcomes or products that are directly under the control of the test player. In many cases, this means that the product must be defined as an intermediate product of the system, not the end product. For instance, if a soldier initiates a missile launch, his task ends in success when he completes the launch sequence correctly and on time (although the missile may not launch because of an electrical problem, much less hit the target). Or, consider a user requirement that the product of a test player's interaction with a software system be a document with a special font or type size that the system does not support. Even though the document fails to include the required font or type size, the test player's performance must be scored a success if he has properly done (i.e., accurately and on time), everything that is directly under his control. The gist of this second point is that all system performance failures are not necessarily due to human performance failures. This distinction must be made to evaluate human performance and software usability, but it does not affect the reporting and scoring of Operational Mission Failures (OMF).

c. Each task trial consists of specific steps or subtasks that allow its process to be analyzed step by step. When it is important to aggregate task performance results, tasks are scripted. At each interface, the set of tasks assigned during the test is either exhaustive or typical of all the tasks intended to be done at the interface (to the extent feasible). In IOTE or later evaluations, tasks are assigned as part of realistic, interactive unit scenarios.

2.3.x.1.2 **Measures of Performance (or Criteria).**[1]

a. Task Completion Rates. [Higher][2]

b. Task Success Rates. [Higher]

c. Task Execution Times. [Lower]

2.3.x.1.3 **Rationale.** A task is effective when the direct outcomes or products of human interactions or subtasks meet minimum user requirements for accuracy (correctness, completeness, or acceptability) and timeliness. Checklists for compliance with minimum user requirements are extracted from Functional Descriptions, Army Regulations, or other format standards.

---

[1] Criteria developed by the evaluator may have only a measure (parameter) specified (p. 42, Enclosure 7, OPTEC Interim Policy Guidance 92-TER, 1 Jul 92). [This means that AOIC may use Measures of Performance (MOP) in lieu of complete criteria.]

Sometimes there are complete criteria for the performance MOPs in this paragraph, but there will normally be no criteria for the other MOPs in this appendix. You must exercise caution when there is a criterion that appears to contain one of the performance MOPs. The criterion may be a training standard rather than an operational performance criterion. It may also confound Task Completion and Task Success.

[2] The notation in brackets, following each MOP in this Appendix, shows which end of the measurement scale normally indicates **greater software usability.** However, this conclusion about each MOP must always be validated **after each test trial**, by a different data source. For example, a person may take more frequent rest-breaks, not because of work avoidance, but because of fatigue unrelated to use of the software interface. The MOPs are marked as follows:

[Higher] =	Higher rate, greater frequency, larger number, or longer time.
[Lower] =	Lower rate, lower frequency, smaller number, or shorter time.

2.3.x.1.4 **Evaluation Approach.** For each software interface, the data on the three measures of performance (MOP) above will be displayed by each separate task assigned during the test. Aggregate MOPs will be calculated across all similar types of tasks done at each software interface (except multimodal Task Execution Times). Task Success Rates will be compared <u>across tasks</u> within each software interface to identify the tasks that are most and least effectively done. They will also be compared <u>across software interfaces</u> to identify the most and least effective software interfaces of the system. Comparisons of the above will also be made with any similar data taken from previous testing of the system. All inferential statistical tests will be done at the .05 level of significance. They will be two-tailed tests, as applicable.

2.3.x.1.5 **Analysis of MOPs and Data Presentations.**

a. <u>Task Completion Rates</u>. **Percent of trials attempted that are completed, per software interface task.** The product or outcome is considered complete by the test player (or supervisor or data collector, as appropriate). The Task Completion Rates will be displayed by each task done at each interface, and aggregated for each interface and for the system as a whole. If there are similar data taken from previous testing of the system, Hotelling's T^2 statistic will be used to see whether there have been significant changes in Task Completion Rates for each software interface and for the system as a whole. The normal distribution (or t Test) will be used if there are no data on multiple tasks.

b. <u>Task Success Rates</u>. **Percent of trials attempted that are completed successfully, per software interface task.** The completed product or outcome is independently judged, by subject matter experts (SME) or analysts, to be an accurate, usable, successful product. The Task Success Rates will be displayed by each task done at each interface, and aggregated for each interface and for the system as a whole. If there are similar data taken from previous testing of the system, Hotelling's T^2 statistic will be used to see whether there have been significant changes in Task Success Rates for each software interface and for the system as a whole. The normal distribution (or t Test) will be used if there are no data on multiple tasks.

c. <u>Task Execution Times</u>. **Median elapsed time for task execution, per software interface task.** Event timing does not include timeouts for breaks or interruptions. Task Execution Time is applicable to all task trials, whether unfinished, completed, unsuccessful, or successful. Task Execution Times for all types of tasks done at an interface are likely to be multimodal. If so, Task Execution Times for different types of tasks done at the same interface will not be aggregated.

2.3.x.2 **Subissue X.2** How well do the system's software interfaces make <u>efficient use of human resources</u>?

2.3.x.2.1 **Subissue X.2.1** How <u>usable or user friendly</u> are the system's software interfaces?

2.3.x.2.1.1 **Scope.** This issue addresses the usability (ease of use or user friendliness) of each major software interface of the system. It is the primary issue that the operational evaluator uses to evaluate software interface usability across the system development cycle, and across systems. To be efficient, producing the result or outcome of the interaction must require a minimum of real-time human resources. Human resources related to software interface usability include the numbers, time, effort, and frustration of users expended to complete a task or interaction. The required operational setting and units of analysis for testing this issue are the same as described in paragraph 2.3.x.1.1.

Usability Profiles
Critical to Longitudinal and Cross-Interface Assessments

This issue, its criteria, and its data collection techniques require special emphasis. Not only is user acceptance an important part of software interface usability, but the MOPs for this issue, especially the Usability Profiles, may be the only adequate data available to the Operational Evaluator during Technical Testing or PM- or contractor-controlled testing. Therefore, this is the primary issue that will allow the Operational Evaluator to evaluate the progress of software interface usability across the system development cycle. For the same reasons, this issue is also most likely to enable the Operational Evaluator to compare software interface usability across two or more interfaces, and even with other systems.

2.3.x.2.1.2 **Measures of Performance.**

a. Interface Usability Ratings (Usability Profile). [Higher]

b. Task Usability Ratings. [Higher]

c. Behavioral Observation Rates:

(1) Rest-Break Rates. [Lower]

(2) Subtask Repetition Rates. [Lower]

(3) Error Rates. [Lower]

(4) On-line Help Access Rates. [Lower]

(5) Documentation Access Rates. [Lower]

(6) Assistance Request Rates. [Lower]

(7) Tutorial Review Rates. [Lower]

(8) Verbal Complaint Rates. [Lower]

(9) Non-Verbal Complaint Rates. [Lower]

d. Test Trials Completed Daily. [Higher]

2.3.x.2.1.3 **Rationale.** To be efficient, the product or outcome of the interaction must be obtainable with a minimum of current human resources. This is enhanced when each software interface is usable. A software interface is usable when acceptable levels of performance are obtained with little expenditure of effort by the user.[1]

2.3.x.2.1.4 **Evaluation Approach.** For each software interface, the data (on all but the Usability Profile) of the MOPs above will be displayed by each separate task assigned during the test. Aggregate MOPs will be calculated across all tasks done at each software interface. The MOPs (except the Usability Profile) will be compared <u>across tasks</u> within each software interface to identify the tasks for which the software is most and least usable within each interface. The MOPs will be compared <u>across software interfaces</u> to identify the most and least usable software interfaces of the system. Comparisons of the above will also be made with any similar data taken from previous testing of the system. All inferential statistical tests will be done at the .05 level of significance. They will be two-tailed tests, as applicable.

2.3.x.2.1.5 **Analysis of MOPs and Data Presentations.**

a. <u>Interface Usability Ratings (Usability Profile)</u>. **Usability ratings of software interactions by test players at the completion of their participation in the test.** There is one Usability Profile for each major software interface. The rating

[1] An efficient software interface is both usable and trainable. These characteristics are related but can have different consequences. A system that is easy to learn may still be unusable because it is inherently awkward. Another system may be usable, but only after a significant investment of training resources. The next issue will address the trainability of software interfaces.

items for any given interface typify all known interactions that will occur at that interface upon fielding of the system. Each Usability Profile consists of mean numerical ratings of human-software interactions in the following categories:

- Data Display Interactions
- Data Entry Interactions
- Error Management Interactions
- Interactive Control Interactions

Detailed data displays will include the mean, standard deviation, sample size, and response rate for each rating item, for each human-software interaction category, and for each software interface. The normal distribution (or t Test) will be used to see whether each mean rating is significantly different from the theoretical neutral point (5.0), of the universal rating scale used to generate the Usability Profile. This will be done only at the level of data displayed in the Usability Profile. The actual significance tests should be done in the evaluation or assessment report by displaying 90% confidence intervals of the mean ratings versus the universal rating scale.

b. Task Usability Ratings. **Self-report ease-of-use ratings of software interactions by test players at the end of each task trial.** They emphasize human-software interactions that are expected to vary from one task repetition to the next. Each type of interaction is given a mean numerical rating. Detailed data displays will include the mean, standard deviation, sample size, and response rate for each ease-of-use interaction item (i.e., each questionnaire item), for each type of task, and for each software interface. For each software interface, mean software interaction ratings will be used to rank two variables by decreasing favorableness toward the software usability of the interface. The first list will show the degree to which doing a task was assisted by using the software interface (i.e., task difficulty using the software system). It will rank tasks by the mean rating for each type of task, taken across all ease-of-use interaction items. The second list will show the degree to which a software interaction was doable, or user friendly, during tasks. It will rank ease-of-use interaction items by the mean rating for each item, taken across all types of tasks.

c. Behavioral Observation Rates. Frequency data (taken during real-user testing), on the behavioral items below yield two MOPs per behavior:

- **Percent of Trials** in which the test player exhibited the behavior
- **Median Frequency** with which the behavior was exhibited

Detailed data displays will show the sample size, percent, and median, low, and high frequency for each MOP, for each type of task, and for each software interface.

(1) Rest-Break Rates. Test player took a mid-task rest-break on own initiative.

(2) Subtask Repetition Rates. Test player repeated a step in the task.

(3) Error Rates. Test player committed an error on a preselected critical step or interaction, by type of error.

(4) On-line Help Access Rates. Test player used on-line help.

(5) Documentation Access Rates. Test player used printed software documentation.

(6) Assistance Request Rates. Test player requested assistance from someone else.

(7) Tutorial Review Rates. Test player reviewed an on-line tutorial.

(8) Verbal Complaint Rates. Test player expressed displeasure vocally.

(9) Non-Verbal Complaint Rates. Test player expressed displeasure physically (e.g., slamming a hand on the table or throwing the mouse).

d. Test Trials Completed Daily. **Median number of task trials completed per test player, per test day.** Detailed data displays will show the sample size and median, low, and high frequency, for each type of task, and for each software interface.

2.3.x.2.2 **Subissue X.2.2** How trainable are the system's software interfaces?

2.3.x.2.2.1 **Scope.** This issue addresses the trainability (ease of learning) of each major software interface of the system. To be efficient, the preparation needed before a task can be assigned to a user must also require a minimum of human resources. Human resources related to software interface trainability include the skill levels, experience, training, and education invested in the preparation of users to interact with the system. The required operational setting and units of analysis for testing this issue are the same as described in paragraph 2.3.x.1.1.

2.3.x.2.2.2 **Measures of Performance.**

a. Slopes of Performance v. Repetitions Curves (Learning Curves).[1] [See Evaluation Approach below]

(1) Task Success Rates.

(2) Task Completion Rates.

(3) Task Execution Times (by task).

(4) Behavioral Observation Rates (by task).

b. Training Tasks Completed Daily. [Higher]

c. Training Time to Qualification. [Lower]

d. Training Trials to Standard. [Lower]

2.3.x.2.2.3 **Rationale.** To be efficient, the product or outcome of the interaction must be obtainable with a minimum expenditure of human resources to prepare users to work with the system. This is enhanced when each software interface is trainable. A software interface is trainable if acceptable levels of performance are achieved quickly or with little practice.

2.3.x.2.2.4 **Evaluation Approach.** For each software interface, the data on the MOPs above will be calculated by each separate task assigned during the test. Aggregate MOPs will be calculated across all tasks done at each software interface. MOPs that can generate a learning curve will be displayed in line graphs with task repetition numbers on the abscissa. These line graphs will be compared to the classical learning curve and to each other to judge whether the slopes of the measured learning curves peak quickly (high trainability), normally (normal trainability), or slowly (low trainability), during the test period. If learning curves are flat during the test period, they will be classified by high or low trainability, as appropriate. The MOPs will be compared across tasks within each software interface to identify the tasks for which the software is most and least trainable within each interface. The MOPs will be compared across software interfaces to identify the most and least trainable software interfaces of the system. Comparisons of the above will also be made with any similar data taken from previous testing of the system. All comparisons of learning curve data will be judgmental. Any inferential statistical tests

[1] When plotted by iteration number or cumulative practice time, performance indices show information similar to the slopes of learning curves, without the requirement to fit curves to data.

for the remaining MOPs will be done at the .05 level of significance. They will be two-tailed tests, as applicable.

2.3.x.2.2.5 **Analysis of MOPs and Data Presentations.**

a. Slopes of Performance v. Repetitions Curves (Learning Curves). When plotted by repetition number or cumulative practice time, the following performance indices show information similar to the slopes of learning curves, without the requirement to fit curves to data. The MOPs in this paragraph will be displayed in line graphs with task repetition numbers on the abscissa. Analysts will compare these line graphs to the classical learning curve and to each other and judge whether each graph shows high, normal, or low trainability.

(1) **Task Success Rates, by repetition number**

(2) **Task Completion Rates, by repetition number**

(3) **Task Execution Times (by task), by repetition number**

(4) **Behavioral Observation Rates (by task), by task repetition number**

b. Training Tasks Completed Daily. **Median number of training tasks completed per trainee, per training or practice day.** Detailed data displays will show the sample size and median, low, and high frequency of the MOP, for each type of task, and for each software interface.

c. Training Time to Qualification. **Median number of training or practice days (or hours) required to reach minimum levels of performance (i.e., training qualification standards), by software interface.** Training can be either instructor- or self-paced. Detailed data displays will show the sample size and median, low, and high frequency of the MOP, for each type of task, and for each software interface.

d. Training Trials to Standard. **Median number of practice trials required to reach minimum levels of performance on specific training tasks.** Training can be either instructor- or self-paced. Detailed data displays will show the sample size and median, low, and high frequency of the MOP, for each type of task, and for each software interface.

APPENDIX B
DATA COLLECTION TOOLS

B.1 **TOOLS FOR ASSESSING USABILITY.** The type of assessment you can do depends upon the type of test event and who controls the test. After you become familiar with the system, the next step is to decide the type of assessment you can do at the next data collection opportunity. For example, early in the development process, a system cannot be formally tested by having users interact with the equipment and then fill out questionnaires. This is because the system has not yet developed to a point where these methods are feasible. Neither the capabilities of the software interfaces, nor the task inventories of job requirements, will be complete. The assessment then would be a checklist assessment of the software usability features included in the design of each software interface. You (the Operational Evaluator, analyst, or observer), will do this by developing and filling out checklists after reviewing the system documentation and prototype design information.

B.1.1 Selection of Data Collection Techniques. Table B.1 (also found as Table 2.2 and discussed in Chapter 2), shows the types of software usability assessments you can do, given the type of test event and who is controlling data collection at the test. Note that this table is very useful for long-range planning. It will help you ensure that all software usability data collected by OPTEC during the system acquisition process, will be complementary. It will also help you do early planning and negotiating with the Program Manager (PM) (or Technical Tester/Evaluator), to get data you might not get if you ask for it at the last minute before a test.

B.1.1.1 Selecting the Type of T&E Activity (Column). To use the table, first select the type of test event from the columns in the table. The second column refers to Technical Test and Evaluation activities (i.e., compliance testing). It also refers to contractor-conducted development or engineering testing. The third column refers to operational test and evaluation activities. You will select only one column that applies to a data collection opportunity.

B.1.1.2 Selecting the Control of Data Collection Tools (Rows). After you select the appropriate column, notice that both rows in the column can apply to your situation. Even when the primary purpose of the test is to allow the PM, Technical Tester/Evaluator, or contractor to collect data, you probably can negotiate with the PM or Technical Tester/Evaluator to have OPTEC personnel collect some data directly, and to have indirect access to data collected by PM, Technical Tester/Evaluator, or contractor personnel. Even in an OPTEC test, there may be PM, Technical Tester/Evaluator, or contractor collected data that you would like to analyze (e.g., contractor "Help Desk" data).

B.1.1.3 <u>Selecting Potential Data Recorders</u>. After you select the appropriate cells of the table, you will focus on the data recorders and the types of data collection tools to use. Depending upon many factors (e.g., budget, time schedule, support personnel), you must choose which options are best for you.

Table B.1 Data Recorders[1] and Data Collection Tools

Type of T&E Activity v. Control of Data Collection Tools

DATA COLLECTION TOOLS DEVELOPED & ADMINISTERED BY:	TECHNICAL TEST & EVALUATION ACTIVITIES (CD, ET, TT, OT, et al.)	USER TESTING & OPERATIONAL TEST & EVALUATION ACTIVITIES (EUT, LUT, IOT, FOT, et al.)
OPTEC PERSONNEL	1. Evaluator/Analyst[1] - Checklist/ Document Review - Checklist/ Planning Meetings - Checklist/ Observation - Checklist/ Hands on (?)[2] - Structured Interviews (?)[2 & 3] 2. Users (?)[2] - Questionnaires 3. Data Collectors/SMEs (?) - Checklist - Questionnaire - Performance Data[3]	1. Evaluator/Analyst[1] - Checklist/ Document Review - Checklist/ Planning Meetings - Checklist/ Observation - Checklist/ Hands on - Structured Interviews[3] 2. Users - Questionnaires 3. Data Collectors/SMEs - Checklist - Questionnaire - Performance Data[3] - SME Judgments 4. Instrumentation - Audit/Keystroke Trail - Video/Audio Data
PM, TT&E, OR CONTRACTOR PERSONNEL	1. Users (?) - Questionnaires 2. Data Collectors (?) - Checklist - Questionnaire - Performance Data - SME Judgments - Structured Interviews[3] 3. Instrumentation (?) - Audit/Keystroke Trail - Video/Audio Data	1. PM/Contractor Collected Information (?) - Checklists, Questionnaires, Structured Interviews, Instrumented data, Help Desk Logs, etc.

[1] Data recorders are the entities that initially record data. An evaluator/analyst is a data recorder when he writes comments and completes checklists. Data recorders in each cell are numbered.

[2] The question mark (?) after some data recorders and data collection tools indicates an area that you will have to negotiate with the PM (or Technical Tester/Evaluator), in order to collect or use this information.

[3] Interview guides for structured interviews should use the same data collection forms as for a questionnaire. The difference is that the evaluator/analyst (or a data collector), records the responses. (Some data recorded by data collectors may be short categorical responses (e.g., "Yes/No" or "Hard/Easy/Neither"), from Test Players. In this case, data collectors will read a short question (and its answers), from a performance data collection form and check the appropriate response block directly on the form.)

However, note that the options listed within each cell are not mutually exclusive. You must always try to get as much as you can of both objective and subjective data on the same test events. You will get your best data on software usability from the performance data and the interview/questionnaire responses of real Test Players doing real tasks.

B.2 EVALUATOR SUPPLIED INFORMATION/CHECKLIST.

B.2.1 Document Review Checklist. As mentioned above, during the early phases of system development it is preferable to use a checklist to collect information on the potential usability of the system interface. This method is used when an actual working system does not yet exist. At this stage the system may consist only of written documentation and paper-copy designs of display screens. The assessment is done by developing and completing one or more checklists. If possible, use a checklist that is unique to each software interface that the system has. To complete the checklists, either use an early prototype of the system, or go through the written documentation and paper-copy screen designs. This will help you predict whether the system will be usable when finally developed. Lists of potential items you can use to develop checklists, along with instructions, are in Appendix C. You can also use checklists to create a structured method for checking system documentation packages. System documentation packages consist of the system requirements, specifications, standards, and other system documentation (such as screen designs). You can use the same lists in Appendix C to develop this checklist.

NOTE: The process of developing checklists will also give you entry into interactions with the contractor(s) and people in the program manager's office (PMO). This can help you learn more about the system's status, progress, and problems than you would otherwise.

B.2.2 Planning Meetings Checklist. When you attend planning meetings, briefings, and system reviews, it is useful to have a checklist handy. The checklist serves as your memory aid (pertinent to software usability), to collect the appropriate information, ask the right questions, and learn the system development status and future testing events. You can develop such a checklist as you go through the process described in this Guide. The important items to be tracked will vary from system to system (and from interface to interface), but this can be a useful tool to find whether system development is progressing. This process will help you obtain the information you need to plan for the best possible assessment. Sources of information that are especially important to the software usability evaluation effort are meetings by the MANPRINT Joint Working Group (MJWG), the Test Integration Working Group (TIWG), the Integrated Logistics Support Management Team (ILSMT), and the System Safety Working Group (SSWG).

B.2.3 Observational Checklist. For this form of assessment, you observe soldiers (or other Government users), interacting with the software. You may be present during system operation, or the test events can be videotaped for later analysis. In either case, you complete a structured checklist of the items, features, and problems you observed. A very useful technique for an observational checklist assessment is to have the Test Players "think aloud" as they work at the software interface. This adds to the information obtained from your own observations. Indeed, if Test Players are properly encouraged to "think aloud," the additional data can be almost like the results of a real-time interview. You can use the same checklist item lists in Appendix C to develop an observational checklist. See Appendix C for more information.

B.2.4 Hands On Checklist. This checklist is similar to the one above except that you complete it yourself, after or during actual use of the system or a system prototype. During the early stages of development, the system may exist to the point that you can sit down and interact with a subset of what the final system will look like. This can provide very useful data about the usability of the final product. You would design a checklist as normal, and fill it out as you work with the early version of the system. If possible, you should work with real tasks, or even scripted tasks to be used in testing. You must keep in mind that system bugs, and other problems, should not necessarily count against the system, since it is still under development. However, it is important to record all bugs that you encounter, because you just may have been the first lucky person to find that particular bug. See Appendix C for more information.

B.3 USER SUPPLIED INFORMATION/QUESTIONNAIRES.

B.3.1 Questionnaires. The questionnaire involves a series of questions presented to a group of software interface users to gather opinions, attitudes, and preferences concerning a system, components of the system, or procedures involved in using the system. Questionnaires usually require a user to rate the component or operation on a scale. Based on these responses, you can find what the Test Players think of the system. The questionnaire is developed from the questionnaire item lists provided in Appendix C. Please see Chapter 2 and Appendix C for more information.

B.3.2 Interviews. Like questionnaires, interviews are conducted with a series of questions developed in advance. These questions are the basis of the interview guide that you should use for a structured interview. Unlike questionnaires, which can be administered to groups, interviews are administered to individual respondents as a structured discussion between the interviewer and the respondent. The interview avoids one of the problems involved with questionnaires. That is total dependence on your ability to identify problem areas where additional information is required. For example, a particular topic could surface as a problem area during the analysis of the questionnaires. However, after the fact, it is too late to discuss the problem with the respondents to gather further information. During an interview, you may identify these problems or trends early in the discussion. You can then pursue it with the respondents to gain deeper insight into user-hostile aspects of the interface. It is a very useful technique to develop the interview guide, including its response formats, just as you would a questionnaire. You would give one copy to the interviewee, but read all the questions, and write all the answers, yourself. Indeed, the only visible difference between a questionnaire and an interview guide may be that the latter requires more intermittent space throughout the data collection form to record your notes. You can use the same lists of questionnaire items in Appendix C to develop interview guides.

B.3.3 User Logs. During certain activities, including tests, it is possible to obtain logs or written information maintained by the Test Player during the operation of the equipment. This is quite common for the roles which require administrative activities such as Preventive Maintenance Checks and Services (PMCS) and system backups. The information contained in these logs can be useful in determining what, if any, difficulties the Test Player encountered while interacting with the system. This information can also be cross-tabulated with other types of data to provide a clearer picture of problem areas.

B.4 SUBJECT MATTER EXPERT (SME) JUDGMENTS.

B.4.1 Quality of the Product or Output of a Test Player's Efforts. It is very important to assess software usability as real users try to do real work at the software interface. Test units and Test Players are normally given unit scenarios and task scripts that simulate on-the-job work requirements that will be done on the full-up system. The **scenario** establishes organizational and situational realism, within which the system is used to do assigned tasks. It is usually written for organizational units or subunits. The **script** provides the specific instructions to an individual or team regarding the task to be done. Subject Matter Experts (SME) can give invaluable job- and task-specific advice and expertise during the design of unit scenarios, task scripts, and data collection sheets and during data reduction. From the appropriate regulations and manuals, they can develop structured scoring systems that measure the degree to which the completed product or outcome of each software interface task is accurate, usable, and successful (by the standards of the user proponent for the system). Such judgments normally go beyond the capabilities and duties of data collectors. However, as part of data reduction, you can use SMEs as expert judges of whether the written products (or other archival, "hard copy" results) of a user task, comply with minimum user requirements. These data will help you assess whether trained users (and using units) can use the system to produce effective, quality results.

B.5 **PERFORMANCE DATA.**

B.5.1 Timeliness. As mentioned above, performance data are a critical part of any good evaluation of system and software usability. Besides expert judgments of the effectiveness of the output produced, the time it takes to produce the desired output is a critical variable. When you analyze Task Completion Times with the quality ratings assigned by the SMEs, the results show a great deal about the usability of the system. For instance, a quality product may be achievable but it may take hours to produce. Therefore, timeliness is an indirect measure of software usability and a direct measure of system suitability.

B.5.2 Accuracy/Quality of Output (Product). See paragraph B.4.1 above.

B.6 **INSTRUMENTED DATA.**

B.6.1 Audit/Keystroke Trails. Depending upon the system being tested, it is possible to automatically capture information even at the level of keystrokes. While this can be useful (depending upon the system, the number of Test Players, and the length of the test), it also generates a tremendous amount of data. Such data can be a prohibitive data reduction burden. If the proper hardware and software tools are not available to reduce this type and amount of data to a usable form, you probably should not go out of your way to collect it. If the correct data reduction tools are available, you probably can use instrumented data to catalog patterns of human (and system) errors that identify particular human-software interface problems or bottlenecks. The levels at which instrumented data are recorded can vary greatly. For some systems only the error messages may be electronically recorded. This too can be useful information when the proper tools exist to aid in the analysis of such data.

B.6.2 Video/Audio Data. Another form of instrumentation which can be useful in the evaluation of human-software interfaces is the recording of Test Player actions and verbal comments. These can be very powerful and irrefutable demonstrations of difficult interfaces. This method is used extensively by "Usability Labs" set up by software developers to find the usability of their software products before they are sold. As mentioned above in paragraph B.2.3, videotaped interaction can be used with observational checklists as a method to collect software usability data. The technique mentioned, of having Test Players "think aloud" as they work, is also very valuable when test events are recorded for later analysis.

B.7 **PM/CONTRACTOR COLLECTED DATA.** If your own data collection opportunities in a test are limited, you may gain useful information from the data collected by the PM or system contractor (also the Technical Tester or Technical Evaluator, especially for non-IMA systems). However, you will probably have to negotiate for access to those data. If you are given convenient access to PMO data, even if you are restricted to observing a PM/contractor controlled test, you probably can influence the data collection plans to fit your needs better. For example, you may even be able to write questionnaire items for the equivalents of any End-of-Task or Post-Test questionnaires that the PM\contractor controlled data collectors will administer. If you work out such plans with the PMO well in advance, you will increase your chances for success. When you try to get the PMO to include your own data requirements in his data collection instruments, it will help if the PMO also needs the data. A word of advice, though. If you ever get to the point of having to justify every one of your questionnaire items (especially in a Usability Profile Questionnaire), emphasize strongly that no single question stands alone. State your intention to develop Usability Profiles from the data. These require that you use a large enough sample of questionnaire items that they are collectively typical (or representative) of all the human-software interactions that will be required at the software interface.

B.7.1 Checklist. During a PM/Contractor controlled test (or demonstration), data are often collected by the contractor. The PM may also collect information directly or indirectly. If you are given convenient access to this information in a usable format, it can provide insight into the system's current usability. Although you did not develop the checklist yourself, it may still indicate some aspects of how well the system operates. The usefulness of this information increases when you are allowed to witness the circumstances under which the data are collected.

B.7.2 Questionnaires. During a PM/Contractor controlled test (or demonstration), questionnaire information may also be collected. This can be by the contractor, the government, or support contractor for the government. This questionnaire may be filled out by software engineers, typical users, atypical users, computer consultants/experts, or government personnel. If you are given convenient access to this information in a usable format, it can provide insight into the system's current usability. Although you did not develop the questionnaire yourself, it may still indicate some aspects of how easy the system is to use. As with the checklist, the usefulness of this information increases when you are allowed to witness the circumstances under which the data are collected.

B.7.3 Documentation. During PM/Contractor testing or demonstrations, documentation about the operation of the system will be collected. This information could be in logs kept by the personnel performing the testing, audit trail information collected by the system, logs kept by support personnel such as maintenance or Help Desk activities, Software Trouble/Problem Reports, and many others. This information can be very useful if you are given access to it in a form where you can readily use it. Large amounts of information are usually collected and you must have access to it (preferably in electronic form) so that it can be analyzed and summarized. Reviewing this documentation after the test can provide tremendous insight into the status and capabilities of the system and its usability.

APPENDIX C
ITEMS FOR CHECKLISTS AND QUESTIONNAIRES

C.1 **TABLES OF CHECKLIST AND QUESTIONNAIRE ITEMS.** This appendix contains two lists of software interface usability items you can use to develop checklists and questionnaires for each of the system's software interfaces.[1] The two lists are in four large tables representing these interaction categories: **Data Display** (Table C.1), **Data Entry** (Table C.2), **Error Management** (Table C.3), and **Interactive Control** (Table C.4). Each table arranges the lists in subcategories, not all of which will apply to every software interface. The first wide box in each table describes the interaction category. The remaining wide boxes describe subcategories. Appendix E is an example End-of-Task Questionnaire and Appendix F is an example Usability Profile (or Post-Test) Questionnaire. Appendix G explains how to reduce the data and gives examples of data displays.

C.2 **EDITING ITEMS.** You should add, delete, and modify items so they will be specific **to each software interface** that you want to assess or evaluate at a given data collection opportunity. The items are written at or below the 11th grade reading level. If you think respondents could misinterpret an item, or the item does not apply to your system as written, you should rewrite it. Eliminate any usability item that clearly does not apply to the software interface you are assessing. The four large tables in Appendix C are available in electronic form, in WordPerfect 5.1 format. To get these files, see the MANPRINT analyst in your Directorate (or in the Technical Integration Division).

C.3 **USABILITY CHECKLISTS.** You will use checklists to make projected estimates of Software Usability during the **early phases** of system development. You will also use them for other purposes, including to supplement data in user testing. The checklist items (always on the **left side** of the table), are derived from the **technical** software usability requirements of MIL-STD 1472D and MIL-STD 1801. Use them as YES/NO items for data collection opportunities that have **no real users** at one or more interfaces. These opportunities normally occur early in system development when the specific interactions required at an interface are not yet well-defined.[2] Since you or an analyst will be the data recorder, you can afford to have more items in a checklist than in a questionnaire. To make scoring easier, you

[1] When a user test is part of the assessment, we recommend that you select the checklist or questionnaire items yourself, then hand them off to the Operational Tester to finalize the data collection instruments for the test.

[2] See Table 2.2 (or Table B.1) for a list of applicable T&E activities and data collection tools.

PREPARING A SOFTWARE USABILITY CHECKLIST

1. Develop one checklist for each interface that was selected.

2. Go to the four long tables in Appendix C. Look at the <u>left column</u> under each of the four categories of interactions.

3. Select all subcategories that apply to the particular interface selected.

4. Select all checklist items that apply to the particular interface selected.

5. Write additional checklist items from your current knowledge of the system and its intended functions, tasks, and interactions.

   The goal is to develop a comprehensive set of checklist items that addresses all expected human-software interactions that will be required (and possible) at the interface.

6. Use the entire list of checklist items that you have selected.

   If that is infeasible, make certain you include each checklist item that covers a critical task. Then use a table of random numbers to select from the pool of remaining items. You should have a total of 15-25 items from each of the four categories.

7. Score each interface in each of the four categories of interactions by tallying the percentage of responses that show a favorable response to the usability and trainability of the interface.

may want to write all checklist items so a YES response is always a favorable response toward the usability of the system.

C.3.1 <u>Scoring of Checklist Results</u>. There will be one checklist for each software interface you assess. Score each checklist by calculating the percentage of checklist item responses that are favorable toward the system. Do this for each of the four interaction categories, and for the aggregate across them. Appendix G shows how to calculate the results and display them.

C.4 **USABILITY QUESTIONNAIRES.** You will use questionnaires to develop Usability Profiles of software interfaces during any phase of system development. The questionnaire items (always on the **right side** of the table), emphasize the dynamic interactions that take place at a software interface. They explore the user's ability to use the software interface effectively. Use them as scaled numerical response items for data collection opportunities that have **<u>real users</u>** at one or more interfaces. These opportunities are likely to occur during the late phases of system development when the specific interactions required at an interface are better defined. Instruct respondents to select "5" for any questionnaire item they believe does not apply to them. (This is also the correct response for someone who is not sure, or does not know how to answer). To make scoring easier, you

PREPARING A SOFTWARE USABILITY QUESTIONNAIRE

1. Develop one questionnaire for each interface that was selected.

2. Go to the four long tables in Appendix C. Look at the <u>right column</u> under each of the four categories of interactions.

3. Select all subcategories that apply to the interface.

4. Select all potential questionnaire items that apply to the interface.

5. Write additional questionnaire items from your current knowledge of the system and its intended functions, tasks, and interactions.

   The goal is to develop a comprehensive set of questionnaire items that addresses all expected human-software interactions that will be required (and possible) at the interface.

6. Circulate the list of interactions to several SMEs for help in finalizing the master list of items.

7. Rewrite potential questionnaire items as necessary.

8. Use the entire list of questionnaire items in the Post-Test Usability Questionnaire for each software interface.

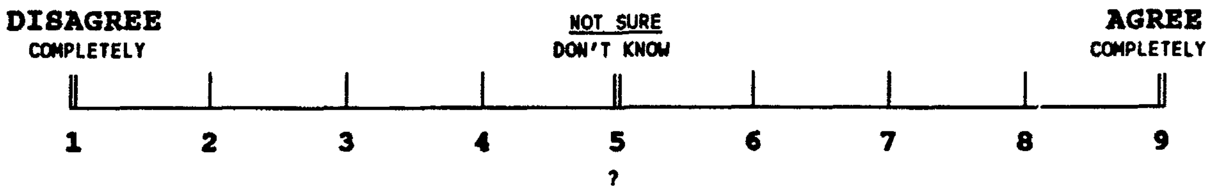
   If that is infeasible, make certain you include each questionnaire item that covers a critical task. Then use a table of random numbers to select from the pool of remaining items. You should have a total of 15-25 items from each of the four categories.

9. Score each interface in each of the four categories of interactions by calculating the mean responses on a dimension of favorableness toward the usability and trainability of the interface.

will want to write all questionnaire items so a "9" response is always the most favorable response toward the usability of the system.

C.4.1 <u>Construction of Questionnaires</u>. The questionnaire items in Tables C.1 - C.4 provide the basis for you to build a comprehensive set of all the human interactions at a software interface that are demanded by the nature of the job and the tasks to be done. Discard items that do not apply to a software interface. Add items as you learn more about the job requirements for the interface. As the system matures, you should derive the set of questionnaire items almost exclusively from job and task requirements at the software interface (<u>not</u> from its technical capabilities). Use the entire list of questionnaire items in the Post-Test Usability Questionnaire for each software interface. Since the maximum number of items a Test Player should have to answer is less than 100 (60-80 may be optimal), it will sometimes be infeasible to use all the items you have developed. In this case, make certain you use each questionnaire item that covers a potential critical task. Then round out the rest of the questionnaire by using a table of random numbers to select from the remaining items in each interaction category.[1] The completed questionnaire item list should have a total of 15-25 items from each of the four categories. When you cannot use the entire list of interaction items, this modified random selection procedure is the best way to ensure that the Usability Questionnaire represents all possible interactions at each software interface.

Numerical Response Scale for Questionnaire Items.



(Select the appropriate whole number)

C.4.2 <u>Scoring of Questionnaire Results (Usability Profiles)</u>. There will be one questionnaire for each software interface that you assess. Each software interface is scored by calculating the mean numerical response on a scale of one to nine. Do this for each questionnaire item, for each of the four interaction categories, and for the aggregate across all interaction categories. When you tabulate these results for all the system's software interfaces, they become the Usability Profiles. Appendix G shows how to display Usability Profiles in Table G.5 and Figure G.1.

C.5 **THE TABLES.** The four large tables follow, one for each interaction category. Each table takes up several pages. To help you keep your place in a table, the pages are renumbered to match the table number and the page of the table. Since you will add and delete many items as you tailor checklists and questionnaires to your system, the items are not numbered.

[1] If you are not certain how to generate random numbers, consult an Operations Research/Systems Analysis (ORSA) analyst or your directorate's MANPRINT analyst.

Table C.1. Potential Checklist and Questionnaire Items -- Data Display Category

Data Display:

This category of human-software interactions encompasses the methods and data formats through which the system conveys information to the user. Most software interfaces use some type of screen display such as Cathode Ray Tubes (CRT), Large Screen Optical Projection, Dot Matrix/Segmented Displays, or Light Emitting Diodes (LEDs).

Display Format:

These items concern how the software interface groups and separates data; whether it uses labels, prompts, and scrolling; and whether it generates hard copies.

Does the system display data in a readily readable and usable format?	It is easy to read and use displayed data.
Does the system display data in familiar units of measure so conversion or translation is unnecessary?	I never have to convert or translate the units of measure for data shown on the display screen.
Does the system display data fields in a naturally occurring order (e.g., chronological, sequential, or logical)?	It is easy to understand lists or sequences of data, because the system displays data fields in a naturally occurring order (e.g., chronological, sequential, or logical).
Does the system display groups of data by separating them with blanks, spacing, lines, or color coding?	The display screen makes good use of blanks, spacing, lines, or color coding to separate groups of data visually.
Does the system group sets of associated data together to illustrate their relationship?	It is easy to tell which groups of data are (or are not) related to other groups of data on the display screen.
Does the system use consistent names for recurring data fields?	It is easy to recognize recurring data fields because they are consistently named.
Does the system use spaces, slashes, or hyphens to break up large data fields (five or more characters long)?	Large data fields are easy to understand because the display screen shows them as separate, smaller chunks of data.
Does the system display data fields that must be compared to each other by placing one above the other?	When necessary, it is easy to compare or contrast key data fields because the display screen places them one above the other.
Does every field in a display have a distinctive label?	It is easy to see what kind of data I am looking at, because every field in a data display has a distinctive label to identify it.
Does every column heading in a display have a distinctive label?	It is easy to see what kind of data I am looking at, because every column heading in a tabular data display has a distinctive label.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.1. Potential Checklist and Questionnaire Items -- Data Display Category (Continued)

Does the system reserve the last lines of the display screen for messages, prompts, and command entries?	I can easily find messages, prompts, and command entries because they are in a predictable place on the display screen (e.g., the last few lines).
Does the page (or screen) label include the current page (or screen) number and the total number of pages (or screens)?	When I work with any kind of document, it is easy to tell from the display screen what page number I'm on, and the total number of pages.
Is windowing, framing, and scrolling consistent across all displays in the system?	The system does not confuse me with windowing, framing, or scrolling characteristics that are inconsistent across data displays.
Does the system display text as it will be printed (e.g., underline, bold characters, and fonts)?	I can easily call up a useful on-screen data display (i.e., a print preview) that shows me exactly how the document will be printed (e.g., underlining, bold characters, and font size).
Does the system permit the user to select the number of pages (all or a partial amount) to be printed?	I can easily select the number of pages that I want to print (i.e., all of them or subsets of them).
Can the user initiate printing by a simple request, rather than through a series of actions?	I can easily initiate printing with a simple request.
Does the system maintain displays of information, signals, warnings, and error messages, long enough for the user to detect them reliably?	The system displays information and signals long enough for me to detect them reliably.
Does the system update multiple displays simultaneously?	Data sets are easy to use because all multiple displays of the same data set are updated simultaneously.
Does the system prevent all noticeable flickering of the display (as in Cathode Ray Tubes)?	The system displays show no annoying flickering that used to be common in older Cathode Ray Tubes.
Does the system use conventional punctuation and spacing?	It is easy to understand displays of textual data because the system uses conventional punctuation and spacing.
Display Content: These items concern multiple displays, abbreviations and acronyms and whether the software interface allows the user to change or edit data.	
Are the displays used in critical steps of a task, easy to read and understand?	It is easy to read and understand the displays that apply to critical task sequences.
Do the display screens provide enough data or information for the user to accomplish the task correctly?	The display screens give me all the data and information I need to accomplish a task.
In multipage displays, are functionally related data items grouped together on one page?	When there are multiscreen displays, the system makes it easy to use functionally related data items, because it groups them together on one screen.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.1. Potential Checklist and Questionnaire Items -- Data Display Category (Continued)

Are codes, abbreviations, and mnemonics in display screens related to standard English and to specific job-related terminology?	It is easy to read and understand the codes, abbreviations, and mnemonics (i.e., memory aids) the system uses. They are all related to standard English or to terminology that applies specifically to my job.
Do words and terms have only one consistent code, abbreviation, or mnemonic?	The system does not confuse me with different abbreviations for the same word.
Does the system use a minimum number of abbreviations?	The system does not confuse me by using many abbreviations.
Does the system code or mark data fields if the user cannot change or edit them (e.g., for security or other reasons)?	One feature that makes the system easy to use is that a data display codes or marks data fields that I am not allowed to change or edit (e.g., for security or other reasons).
Display Coding: These items concern the software interface's use of blinking/flashing, underlining, brightness, color, and symbols when displaying text.	
Is the variability of data displays adequate to allow the user to distinguish between items of information and to attract the user's attention to changes in the status of system processing?	The variability of data displays helps me distinguish between items of information. The variability of data displays adequately attracts my attention to changes in the status of system processing.
Does the system use display coding that does not reduce legibility or increase processing time?	The display of coded data fields does not slow me down, because they are easy to read and understand.
When displayed alphanumeric codes combine letters and numbers, are characters of each type grouped together (e.g., ABC111 not A1B1C1)?	It is easy to read displays of alphanumeric codes of combined letters and numbers, because the system groups characters of each type together (e.g., ABC111 not A1B1C1).
Does the system display alphanumeric codes all in upper case?	It is easy to read alphanumeric codes, because the system displays them in upper case (i.e., capitalized).
Do display screens use no more than two flash or blink rates simultaneously?	The system does not display a confusing number of different flash rates or blink rates on the same display screen.
If the user must read critical data, does a line under the data blink, not the data entry itself?	If the system uses blinking to draw my attention to a critical data field, a line under the data field blinks, not the entire data field.
Does the interface use no more than two levels of brightness on the same display screen?	The system does not display a confusing number of different levels of brightness on the same display screen.
Does underlining sufficiently attract the user's attention to the underlined item?	The system's use of underlining is sufficient to attract my attention to the underlined item.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.1. Potential Checklist and Questionnaire Items -- Data Display Category (Continued)

Are the system symbols in general use (i.e., are they conventional, customary, well-known, and familiar)?	The symbols used by the system are easy to understand because they are in general use (i.e., they are conventional, customary, well-known, and familiar).
Do special symbols have only one purpose and definition?	The system does not confuse me by using special symbols that have more than one purpose or more than one definition.
Do the colors in the system displays use common conventions?	The colors in the system displays appear to use common conventions.
Does color make the display easy to read and not strain eyes, obscure the display or make the display illegible?	Displays that use colors are easy to read (without causing eyestrain, obscuring the display, or making the display illegible).
Do system displays use color consistently?	All display screens that use color, use it in similar, consistent ways.
Do the shapes of screen or display features conform to conventional, accepted usage?	The shapes of screen features or display features appear to conform to conventional, accepted usage.
Does the system use reverse video (dark characters on a bright background) only to code critical items of information?	The system appears to use reverse video (i.e., dark characters on a bright background) only to code critical items of information.
Does the system change the size of a data item to call attention to it, only in uncrowded, uncluttered displays?	Display screens use different font sizes to call attention to a data item, only in uncluttered displays.
Do display screens of alphanumeric data have no more than three levels of size changes or differences?	Display screens do not use an excessive number of different font sizes to call attention to data items.
Does the system use auditory signals only to supplement visual displays or to cue the user to needed actions?	The system uses auditory signals only to supplement visual displays (or to let me know that I have to take an action).

Dynamic Displays:

A dynamic display is one in which the basic format of the screen remains frozen, but data fields are continuously updated until the processing sequence is complete. These items concern whether the user can freeze the display any time.

In a dynamic display, are changing alphanumeric values updated fast enough to be considered real-time, but slowly enough for the user to read them reliably?	[A dynamic display is one in which the basic format of the screen remains frozen, but data fields are continuously updated until the processing sequence is complete.] When I look at dynamic displays, it is easy to read and understand the rapidly changing alphanumeric values.
In a dynamic display, can the user modify the rate of changing displayed values, depending on the use of the information?	I can easily modify the rate at which selected key values change in dynamic data displays.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.1. Potential Checklist and Questionnaire Items -- Data Display Category (Continued)

In a dynamic display, does the system provide an alert if it detects some significant (but not displayed) change of data when in the freeze mode?	After I freeze a dynamic display screen, the system still provides an adequate alert if it detects some significant (but not displayed) change of data.
In a dynamic display, does the system remain in the freeze mode until commanded otherwise?	After I freeze a dynamic display screen, the system remains in the freeze mode until I command it otherwise.
In a dynamic display, will the system automatically resume updating the display at the current real-time point after a freeze, unless otherwise specified? [Similar to the way a stopwatch continues to run after the user reads a lap time or split time.]	After I freeze/unfreeze a dynamic display screen, the system automatically resumes updating the display at the current real-time point, unless I command it otherwise. [Similar to the way a stopwatch continues to run after the user reads a lap time or split time.]
In a dynamic display, does the display indicate when the system is in the freeze mode? [Similar to the way a stopwatch lets the user know that a lap time or split time is being displayed when the user freezes the display to read it.]	After I freeze a dynamic display screen, the system display gives a noticeable indication that it's in the freeze mode. [Similar to the way a stopwatch lets the user know that a lap time or split time is being displayed when the user freezes the display to read it.]
Display Suppression: These items concern whether the operator can blank the screen during normal operations.	
When the user has the option to blank out or suppress certain data from being displayed, does the display provide an indication (e.g., a label) when data suppression has been selected?	When I have the option to blank out or suppress certain data from being displayed, the display gives a noticeable indication (e.g., a label) that I have selected data suppression.
When the user has the option to blank out or suppress certain data from being displayed, does the system provide an alert when suppressed data changes significantly?	When I have the option to blank out or suppress certain data from being displayed, the system provides an adequate alert when there is a significant change in the suppressed data.
When the user has the option to blank out or suppress certain data from being displayed, does the system provide an easy method to restore suppressed data to the display?	When I have the option to blank out or suppress certain data from being displayed, the system provides an easy way to restore the suppressed data to the display.
Tabular Displays: These items concern how the software interface uses tables to display data.	
Does the system display similar data in the same location on all related data tables (e.g., Columns 1 and 2 are PART# and QUANTITY in all tables)?	It helps to find data in tabular displays, because similar data are located in the same position on all data tables (e.g., Columns 1 and 2 are PART# and QUANTITY in all tables).

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.1. Potential Checklist and Questionnaire Items -- Data Display Category (Continued)

If there is an order to tabular data, are the data displayed in that order (e.g., data in the column is in ascending or alphabetical order)?	It helps to find data in tabular displays, because data items are displayed in order (e.g., data in the column is in ascending or alphabetical order, etc.).
Does the system arrange lists in some logical order (e.g., chronological, alphabetical, sequential, functional, or by importance)?	It helps to find data in tabular displays, because lists of data are arranged in the most logical order (e.g., chronological, alphabetical, sequential, functional, by importance, etc.).
Is each item in a list uniquely numbered or lettered?	When the system displays lists of data, each item in the list is numbered or lettered uniquely.
Does each item in a list start on a new line?	When the system displays lists of data, each item in the list starts on a new line.
If lists extend beyond one screen, is the last line of one screen the first line on the next screen (e.g., PAGE DOWN makes the last item on one screen appear as the first item on the new screen)?	When lists of data extend beyond one page, the last line of one page is on the first line of the next page (e.g., page down makes the last item on one page appear as the first item on the new page).
Does the system use Arabic numerals instead of Roman numerals when numbering items in a list (e.g., 1, 2, 3, not I, II, III)?	When the system displays lists of data, it never uses Roman numerals to number items in a list (i.e., it never uses I, II, III, etc.).
Does the item numbering in a list start with one, instead of zero?	When the system displays lists of data, it never starts numbering the data items with a zero.
Where no conventional punctuation schemes exist (e.g., separation symbols as in $1,001.99), is a space used after every third or forth digit of a long numerical entry (e.g., 176 234 567 9876)?	Where no conventional punctuation schemes exist (e.g., separation symbols as in $1,001.99), the system uses a space after every third or forth digit of a long numerical entry (e.g., 176 234 567 9876).
Does the system use distinctive labels for rows and columns in tabular displays?	The rows and columns in tabular data displays are distinctively labeled.
Do tabular displays use consistent units of measure (e.g., feet and meters are not used in the same table)?	Tabular data displays always use consistent units of measure (e.g., feet and meters are not used in the same table).
Is column spacing uniform and consistent within a table, and from one table to another?	The column spacing is consistent and uniform, within a tabular data display, and from one table to another.
Does the system separate columns in a table by at least three spaces?	The separation between columns in tabular data displays makes the data easy to read.

Graphical Displays:

These items concern how the software interface uses graphics to display data.

Does the system use standard graphics and symbols?	The system uses standard symbols and graphics in graphical displays.
Do recurring data within the system's graphical displays have consistent names?	Graphical displays use consistent names for recurring data fields.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.1. Potential Checklist and Questionnaire Items -- Data Display Category (Continued)

Do recurring data within the system's graphical displays have consistent relative positions in the display?	Graphical displays use consistent relative positions on the screen to display recurring data fields.
Do scales start at zero, except where this would be inappropriate for the function involved?	The scales in graphical charts start at zero, except where this is inappropriate for the mathematical function involved.
When data must be compared across a series of charts, do the charts use the same scale?	When I compare data across a series of similar charts, I find that the charts use the same scales.
Does each graphical display use only one scale on each axis?	Scaled charts never have more than one scale on each axis.
Are the graphical displays' grid lines unobtrusive so they do not obscure data elements?	The graphical display's grid lines were unobtrusive (i.e., they did not obscure data elements).
Is the method of saving and retrieving graphical displays easy and convenient?	It is easy and convenient to save and retrieve graphical displays.
Can filenames be designated for storing graphics data?	It is easy to use filenames to store graphical data.
Does the system automatically scale data to fit the graphical display?	When I make a scaled, graphical display, the system automatically scales the data to fit the display.
Does the system provide templates, tracing techniques, stored forms or other aids to draw figures?	The system provides templates, tracing techniques, stored forms, and other aids to help me draw figures.
Does the system provide a method of changing the size of any selected element on the display?	When I work with a graphical display, it is easy to change the size of any selected element on the display.
When two curves or sets of data must be compared, does the system provide a separate graphical display or plot of the <u>difference</u> scores between the two sets of data?	When I must compare two curves or sets of data, the system provides a separate graphical display or plot of the <u>difference</u> scores, not just plots of two separate curves.
In its graphical displays, does the system provide a reference, baseline, or significance level so the user can compare critical values?	In its graphical displays, the system provides a reference, baseline, or significance level so I can easily compare critical values.
In its graphical displays, does the system provide the actual numeric values when precise reading of the data is required?	In its graphical displays, the system provides the actual numeric values when precise reading of the data is required.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.1. Potential Checklist and Questionnaire Items -- Data Display Category (Continued)

Audio Displays:

These items concern how the software interface uses audio signals, such as beeps and buzzers.

Does the system use audio displays when visual displays are overburdened or when needed to cue, alert, warn, or provide feedback to the user after control actuation, data entry, or completion of timing cycles and sequences?	The system makes effective use of audio displays (e.g., when visual displays are overburdened or when needed to cue, alert, warn, or provide feedback to the user after control actuation, data entry, or completion of timing cycles and sequences).
Does the audio signal increase the probability of detecting the triggering event?	The system makes effective use of audio signals to increase the probability that I will detect certain triggering events.
Does the audio warning signal consist of an alerting signal and an identifying or action signal?	Audio warning signals always consist of an alerting signal and a signal that identifies what action to take.
Are audio caution signals readily distinguishable from warning signals?	Audio caution signals are readily distinguishable from warning signals.
Does the system use audio caution signals only to signify conditions requiring awareness, but not necessarily immediate action?	The system uses audio caution signals only to signify conditions requiring awareness, but not necessarily immediate action.
When no longer applicable, can audio signals be terminated by the system or by manual cancellation?	When audio signals are no longer applicable, they are easily stopped either automatically or manually.
Does the selected frequency band of audio signals differ from the most intense background frequencies at the work site?	It is easy to distinguish between the system's audio signals and other background noises at my work site.
Can audio signals be easily heard?	I can easily hear the system's audio signals.
Are the various audio signals used by the system easily distinguishable?	It is easy to distinguish between the various audio signals that the system uses.
Can the user define the settings for audio alarms?	It is easy to define the settings for audio alarms.
Were there any instances in which acknowledging or terminating an audio alarm decreased speed or accuracy in reacting to the alerting situation?	Acknowledging or stopping an alarm does not affect the speed or accuracy of my reaction to an alerting situation.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.2. Potential Checklist and Questionnaire Items -- Data Entry Category

Data Entry:

This category of human-software interactions encompasses the methods and data formats through which the user conveys information to the system. These methods include: keyboards, keypads, joysticks, lightpens, touch screens, trackballs, mice, grid pads, optical character recognition (scanners or page readers), and voice input.

General:

These items concern data entry in general. They include whether the software interface allows the user to change data, to control the data entry rate, to use a minimum number of keystrokes, and to draw upon data that is already in the system without having to reenter it manually.

Does the system automatically justify data with the decimal system point, the left margin, or the right margin, depending on the type of data?	The system automatically justifies data as I enter it (with the decimal system point, the left margin, or the right margin, depending on the type of data).
Do data entry functions require a minimum of input actions and memorization by the user?	The data entry functions require me to do a minimum of input actions and memorization.
Is an explicit action (such as depressing a DELETE key) required before the system will process a data deletion or cancellation?	The system requires me to do an explicit action (such as depressing a DELETE key) before it will process a data deletion or cancellation.
When the user changes a data value that is not currently displayed, does the system display the old data value and ask for confirmation of the change?	When I change a data value that is not currently displayed, the system displays the old data value and asks for confirmation of the change.
Do all keyboard entries appear on the display (except passwords, strategic information or other secure entries)?	All keyboard entries appear on the display as I enter them (except passwords, strategic information, or other secure entries).
Can the user enter all parts of a single data entry using one method (i.e., a single data entry does not require use of more than one data entry device such as keyboard, numeric pad, trackball, joystick, touch sensitive screen, or mouse)?	I can enter all parts of a single data entry using one method (i.e., a single data entry does not require use of more than one data entry device such as keyboard, numeric pad, trackball, joystick, touch sensitive screen, or mouse).
ɒo data entry labels provide cuing for required data formats (e.g., DATE (MM/DD/YY):__/__/__)?	Data entry labels provide adequate cuing for required data entry formats (e.g., DATE (MM/DD/YY): __/__/__).
Can the user manually control the data entry rate?	It is easy to control the data entry rate manually.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.2. Potential Checklist and Questionnaire Items -- Data Entry Category (Continued)

Is data entry paced by the user, not by the system?	I paced the rate of data entry myself (i.e., I was not continually trying to catch up with the system).
Must there be an explicit user action before the system will accept a data entry?	I must make a simple, explicit action before the system will accept my data entry.
Does the system require only a minimum number of keystrokes to complete a transaction (i.e., there are no keystrokes that could be eliminated)?	The system requires only a minimum number of keystrokes to complete a transaction (i.e., there are no keystrokes that could be eliminated).
Can the system access data that exist in connected devices, systems, or networks? [The purpose is to minimize manual reentry of data that already resides somewhere in the system.]	It is simple and easy for the system to access data that exist in connected devices, systems, or networks (i.e., I do not have to reenter manually, data that already resides somewhere in the system).
Position Designation [Cursor]: These items concern whether the cursor has a home position (e.g., upper left-hand corner of the screen), the user can move the cursor easily, and the cursor always stays within the screen's borders.	
Does the system provide a method of expanding the display (zooming in) to make the positioning of cursors easier and more precise?	The system provides a simple, easy way to expand the display (i.e., zoom in), to make it easier and more precise to position cursors.
Are cursors readily distinguishable from other displayed items (e.g., data entry, labels, and graphics symbols)?	It was easy to distinguish the cursor from other displayed items (e.g., data entry, labels, graphics symbols, etc.).
Can all data be seen clearly through or around the cursor (i.e., the cursor does not obscure the display)?	I can see clearly through or around the cursor (i.e., the cursor does not obscure the display).
When the user selects from among displayed alternatives, as in a menu, is the selected item made more obvious by some type of highlighting?	When I select from among displayed alternatives (e.g., from a menu), the item I select becomes more obvious by some type of highlighting.
Does the cursor wrap around to the beginning or ending of an option list or menu?	The cursor wraps around to the beginning or ending of an option list or menu.
Does the cursor have a consistent home position across all displays (i.e., the cursor will appear at a home position when a display first appears or when it changes)?	The cursor has a consistent home position across all displays (i.e., the cursor appears at a home position when a display first appears or when it changes).
Do cursor controls allow the user to move the cursor quickly and place it accurately?	It is easy for me to move the cursor quickly and place it accurately.
Does the cursor remain stationary on the display until it is moved or repositioned? [The movement can be done either manually by the user, or automatically by the system because of processing changes, display changes, or system status changes.]	The cursor remains stationary on the display until it is moved or repositioned. [The movement can be done either manually by the user, or automatically by the system because of processing changes, display changes, or system status changes.]

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.2. Potential Checklist and Questionnaire Items -- Data Entry Category (Continued)

Does the system provide a simple means to move the cursor from one field to another (e.g., the cursor moves automatically when a data entry is made or with one stroke of the TAB key)?	The system provides a simple means to move the cursor from one field to another (e.g., the cursor moves automatically when I make a data entry or strike the TAB key).
Can the user adjust the sensitivity of the cursor movement to the movement of the cursor control device (i.e., can he adjust the settings so fast movements are less precise and slow movements more precise)?	I can easily adjust the sensitivity of the cursor movement to the movement of the cursor control device (i.e., I can adjust the settings so fast movements are less precise and slow movements more precise).
Is a cross-hair or fine point cursor provided for tasks requiring fine positioning accuracy?	The system gives me a cross-hair or fine point cursor for tasks that require fine positioning accuracy.
Is the cursor prohibited from moving beyond the display boundaries?	The system prevents me from moving the cursor beyond the display boundaries.
Is an action, separate from cursor positioning, required to cause actual entry or activation of a designated position?	When I use the cursor, I must make an explicit action, separate from positioning the cursor, to cause the system to act on my selection.
Are cursor keys located on the same keypad or keyboard that is used for keyboard entry (i.e., cursor control keys are not on a separate keypad)?	Cursor control keys are easy to use because they are located on the same keypad or keyboard that is used for other keyboard entries.

Data Forms:

These items concern whether the software interface allows the user to enter data as if he were filling in a form.

When entering data on a formatted page or screen, are multiple data entry fields separated by a blank space, line, or other designation?	When I enter data on a formatted page or screen, the system adequately separates multiple data entries by a blank space, line, or other designation.
When entering data on a formatted page or screen, are the field sizes indicated (e.g., by underscoring or other size designation)?	When I enter data on a formatted page or screen, the system adequately shows the field sizes (e.g., by underscoring or other size designation).
When entering data on a formatted page or screen, does the system produce a signal when it is ready to accept entries? [Examples can be a highlighted entry field or a colon, followed by a space.]	When I enter data on a formatted page or screen, the system produces a noticeable signal when it is ready to accept entries. [Examples can be a highlighted entry field or a colon, followed by a space.]
When entering data on a formatted page or screen, are related items grouped together?	When I enter data on a formatted page or screen, I find that related items are grouped together.
When entering data on a formatted page or screen, does the system use a standard input form?	The system uses standard forms or formats for data entry.
When entering data on a formatted page or screen, are required fields distinguished from optional fields?	When I enter data on a formatted page or screen, it is easy for me to distinguish required fields from optional fields.
When entering data on a formatted page or screen, does the system provide a prompt, warning, or alert when required data has not been input?	When I enter data on a formatted page or screen, the system provides an adequate prompt, warning, or alert if I have not entered required data.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.2. Potential Checklist and Questionnaire Items -- Data Entry Category (Continued)

When entering data on a formatted page or screen, does the system allow the user to skip some data field entries until later (i.e., to complete data fields out of order if necessary)?	When I enter data on a formatted page or screen, the system allows me to skip some data field entries until later (i.e., to complete data fields out of order if necessary).
Other Data Processing: These items concern whether software interface data entry formats are consistent with the formats of the source data to be entered.	
Are word choices, formats, and styles of display screens consistent with the user proponent's requirements for data entry and control of data?	Word choices, formats, and styles of display screens are consistent with the requirements of my job for data entry and control of data.
When data are entered from a source document, does the data entry display match the format of the source document? [Forms must be the same. Fields that will not be entered do not have to appear on the display screen. However, a corresponding space must be left for them on the display screen.]	When I enter data from a source document, the data entry display clearly matches the format of the source document.
Hardware Control Methods: These items concern the consistency and ease of use of software interface controls such a keyboard, function keys, numeric keypad, light pen, mouse, joystick, trackball, touch panel/screen, optical character recognition device, or voice input. [The keyboard has several types of keys. They are fixed single-function keys (e.g., BACKSPACE) and fixed multifunction keys (e.g., upper and lower case alphabet keys) and variable-function or programmable keys (e.g., F-keys).]	
Does pressing a left arrow key result in the cursor moving left (and similarly with right, up, and down)?	When I press the left arrow key, the cursor moves left, exactly as I expect it to move (and similarly with right, up, and down).
Is use of second-function keys (e.g., SHIFT, CTRL, or ALT), minimized for data entry?	I rarely have to use second-function keys (e.g., SHIFT, CTRL, or ALT), for data entry.
Do function keys associated with DELETE, or other radical changes, have some type of safety feature (such as double-stroke activation or a confirmation prompt)?	Function keys associated with DELETE (or other radical changes), have some type of easy-to-use safety feature (such as double-stroke activation or a confirmation prompt), to prevent me from inadvertently erasing data that I want to keep.
Does the system use fixed-function keys for time-critical, error-critical, or frequently used control option inputs?	The system makes effective use of fixed-function keys for time-critical, error-critical, or frequently used control option inputs.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.2. Potential Checklist and Questionnaire Items -- Data Entry Category (Continued)

Is the operation of the variable-function keys standard throughout the system?	The operation of the variable-function keys (i.e., F1 - F12), is sufficiently standardized throughout the system.
Do definitions of fixed-function keys remain constant during use?	The functions of fixed-function keys are constant throughout the system.
Does the system temporarily disable fixed-function keys when their function is invalid?	The system temporarily disables fixed-function keys when their function is invalid for a particular software program or step.
Do fixed-function keys require only a single trigger action (except delete or radical changes)?	Fixed-function keys require me to make only a single triggering action (except deletions or radical changes).
Does the system provide the status of the fixed multifunction keys when the effect of the keys varies (e.g., a NUM LOCK indicator)?	The system provides an effective status marker for a fixed multifunction key when the effect of the key varies (e.g., a NUM LOCK indicator).
Does the system provide a visual alert when variable-function keys are reprogrammed or turned off (to signify that the standard function is not currently accessible)?	The system provides an effective visual alert when variable-function keys are reprogrammed or turned off (to signify that the standard function is not currently accessible).
Can initial default functions of variable-function keys be restored through a single action?	It is easy to restore the initial default functions of variable-function keys.
Does the numeric keypad increase efficiency?	It was easy to use the numeric keypad to increase my efficiency.
Does the lightpen have a separate, discrete, manually controllable activating/deactivating mechanism?	The lightpen has a separate, discrete, manually controllable activating/deactivating mechanism that is easy to operate.
Does the lightpen project an illuminated circle onto the display screen?	The lightpen projects a readily distinguishable illuminated circle onto the display screen.
Does moving the joystick left result in the cursor moving left (similarly for right, up, and down)?	When I move the joystick left, the cursor moves left, exactly as I expect it to move (similarly for right, up, and down).
Can the user adjust the sensitivity of the cursor movement to the movement of the joystick?	I can easily adjust the sensitivity of the cursor movement to the movement of the joystick.
Does moving the mouse left result in the cursor moving left (similarly for right, up, and down)?	When I move the mouse left, the cursor moves left, exactly as I expect it to move (similarly for right, up, and down).
Can the user adjust the sensitivity of the cursor movement to the movement of the mouse?	I can easily adjust the sensitivity of the cursor movement to the movement of the mouse.
When moved in strictly the x or y direction alone, is there no apparent cursor movement in the other direction?	When I move the cursor strictly in the x or y direction alone, there is no apparent cursor movement along the other axis.
Does moving the trackball left result in the cursor moving left (similarly for right, up, and down)?	When I move the trackball to the left, the cursor moves left, exactly as I expect it to move (similarly for right, up, and down).
Can the user adjust the sensitivity of the cursor movement to the movement of the trackball?	I can easily adjust the sensitivity of the cursor movement to the movement of the trackball.
Are the control features and display features on touch screens easy to use and understand?	The touch screens are understandable and easy to use.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.2. Potential Checklist and Questionnaire Items -- Data Entry Category (Continued)

Are data entered through optical readers or scanners adequate, accurate, and consistent?	I can easily enter data adequately, accurately, and consistently using optical readers or scanners.
Is voice input used when manual and visual performance are constrained?	The system allows me to use voice input easily when a task is so demanding that it constrains my manual and visual performance.
Is acceptance of vocal input by the system adequate, accurate and consistent?	The system accepts my voice input adequately, accurately, and consistently.
Text/Program Editing: These items concern the ease of use of any text editor that is part of the software interface.	
Is the text editor easy to use?	The text editor is very easy to use.
Does the system provide an easy means to move to the top (i.e., head or beginning) and bottom (i.e., foot or end) of the file?	I can easily move to the top (i.e., head or beginning) and bottom (i.e., foot or end) of a file or document.
Does the text editor provide editing commands (e.g., move, copy, delete)?	The editing commands that the system provides (e.g., move, copy, delete, etc.), are simple and easy to use.
Is specified and selected text (i.e., blocked text), highlighted to show its boundaries?	It is easy to see the boundaries of text I have specified or selected (i.e., blocked), because the text is highlighted.
Can the system search both backward and forward for a specified set of data?	It is simple and easy to search a file or document (both forward and backward) to find a specific character string or data item.
Does the user have the option to continue a search past the first occurrence, to the next?	When searching for a specific character string or data item, it is easy to select an option to have the system continue the search past the first occurrence, to the next.
Does the system provide an automatic line break (carriage return) when text entries reach the right margin?	The system provides an automatic line break (carriage return) when text entries reach the right margin.
Are the predefined or default formats correct and easily accessible?	The predefined or default formats are correct and easy to access.
Can frequently used, manually entered formats be stored for later use?	It is easy to store frequently used, manually entered formats for later recall and use.
Can frequently used text segments be stored for later recall and use (e.g., signature blocks, organizational names, office symbols, call signs, or coordinates)?	It is easy to store frequently used text segments for later recall and use (e.g., signature blocks, organizational names, office symbols, call signs, or coordinates).

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.3. Potential Checklist and Questionnaire Items -- Error Management Category

Error Management:

This category of human-software interactions encompasses the methods by which the software interface provides user help, security, data protection, and system protection. These methods include recovery from errors, avoidance of errors, protection of data, security of data, documentation (such as help screens), system records, system aids, and feedback.

General:

These items concern the ease of use and effectiveness of data security functions at the software interface, including error detection, correction, prevention, recovery, description, advice, or help functions.

Do the system's data protection and error management measures work well?	The system's data protection and error management measures work very well.
Does the system provide warning messages or alarm signals for potential threats to data security?	The system provides effective warning messages or alarm signals when there are potential threats to data security.
Does the system display the security classification level for the data in each display?	When I work with classified data, every display screen has a readily distinguishable label showing the security classification level of the data.
Does the system require confirmation of critical entries before carrying out the action?	The system requires me to confirm critical entries before it carries out the action.
Are error messages appropriate for the training received?	Error messages are appropriate for the level of training I received.
Can the user make the system backtrack to previous steps for error correction or change?	It is easy to make the system backtrack to previous steps so I can correct errors or make changes.
Do spelling or other common errors produce valid, but unwanted transactions or processes?	Minor errors on my part (e.g., spelling or punctuation) do not cause the system to produce valid, but unwanted transactions or processes.
Does the system display and highlight for confirmation, any corrections that the system makes to commands, values, and spellings?	The system adequately displays, and highlights for my confirmation, any corrections that it makes to commands, values, and spellings that I enter.

Multiuser Systems:

These items concern the ease and effectiveness of maintaining data and system integrity in multiuser systems.

↑ **CHECKLIST ITEMS**

↑ **QUESTIONNAIRE ITEMS**

Table C.3. Potential Checklist and Questionnaire Items -- Error Management Category (Continued)

Is there any display, message, or procedure that obscures user commands or system instructions, or creates other conditions contributing to error?	There are no displays, messages, or procedures that obscure my commands (or the system's instructions), or otherwise cause me to make an error.
Does the system prevent interference and disruption during multiperson use?	The system appears to handle multiperson use very well by preventing interference and disruption between users.
Does the software interface preempt system operations only when mission survival depends upon preemption?	During multiperson use of the system, I can easily override system operations (but only to prevent a lockup or shutdown of the main system or server).
When the system preempts system operations, does it resume at the point of interference without losing information?	When the main system or server overrides operations at my software interface, and the problem is corrected, it resumes at the point of override without losing any of my data.
User Identification: These items concern the ease of use and effectiveness of any log-on, log-off, and password procedures used at the software interface.	
Does the system use a password or other log-on code to limit a person's access to the system and a person's ability to change system privileges?	It is easy to use the password (or other log-on code) that the system has to limit my access to the system and my ability to change system privileges.
Does the system prevent bypassing or omitting the log-on procedure?	The system works well to prevent me from taking short cuts or omitting the log-on procedure.
Do log-on prompts appear automatically on the display after turning the system ON, without requiring any other action by the user?	Using the system is simple because log-on prompts appear automatically on the display after the system turns ON, without requiring any other action by me or someone else.
When the user enters his password, does keyed entry have either no displayed response or only special characters (like "*") displayed for each character pressed?	When I enter my password, the keyed entry displays no response (or displays only special characters, like "*"), for each character I enter.
Can personnel select their own password(s) (except access to classified data)?	It is easy for me to select my own password(s) (except access to classified data).
Does the system require changing passwords periodically?	The system periodically makes me change my passwords.
Does the system prompt the user for confirmation before executing log-off?	I seldom lose any data I want to keep s because the system always prompts me for confirmation before it accepts my log-off commands.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.3. Potential Checklist and Questionnaire Items -- Error Management Category (Continued)

Data Access: These items concern the ease of use and effectiveness of user functions at the software interface that are designed for the security of protected or classified data.	
Does the system maintain control over and disallow editing and reformatting of protected or classified data?	I am not able to edit or reformat protected or classified data unless I have been granted the appropriate privileges.
Is a means provided for the user to suppress temporarily the current display (i.e., a screen blanker for instances when classified or protected information is displayed and the user's privacy is threatened)?	I can easily activate and deactivate a screen blanker for instances when classified or protected information is displayed and my privacy is threatened by other people near my workstation.
Does the system require the entry of a password before redisplaying a suppressed display and permitting work to resume?	When I'm working with classified or protected data, maintaining privacy at my workstation is a lot simpler because I have to enter a password before I can deactivate the screen blanker and resume work.
Data Entry/Change: These items concern the ease of use and effectiveness of system functions that check data for errors or validate data upon entry.	
Does the system check data entries for correct format, legal value, and range of values?	When I enter formatted data, the system does a good job of checking my data entries for correct format, legal value, and range of values.
Does the system check data entries for completeness of entry?	When I enter formatted data, the system does a good job of checking my data entries for completeness.
Were there any instances where erroneous or invalid entries were accepted by the system?	If I make an inadvertent error while entering formatted data, the system does a good job of rejecting erroneous or invalid entries.
Does the system produce an advisory message or alarm when the user attempts to change controlled items or protected fields?	The system produces an effective advisory message or alarm if I try to change controlled items or protected fields.
Does the system prompt the user to correct command entries that are not recognized or that are inappropriate?	The system does a good job of prompting me to correct command entries that it does not recognize or that are inappropriate.
How clear are the corrective action descriptions that appear in prompting messages?	The system messages that describe corrective actions for a problem are clear and easy to follow.
↑ **CHECKLIST ITEMS**	↑ **QUESTIONNAIRE ITEMS**

Table C.3. Potential Checklist and Questionnaire Items -- Error Management Category (Continued)

Can errors in a command stack be corrected and the stack salvaged when errors are detected in a command stack? [A command stack is a continuous sequence of commands.]	[A command stack is a continuous sequence of commands.] I can easily correct errors in a command stack and salvage the stack when the system detects errors in a command stack.
Does the system display the computer-detected error and the error message continuously until the error is corrected?	When the system detects an error and displays an error message, it continues to display the error message until the error is corrected.
Loss Prevention: These items concern the system's ability to prevent data from being lost during log-off or system failure.	
Before it accepts log-off commands, does the system check pending transactions to learn whether data will be lost?	The system does an effective job of checking pending transactions (to find whether data will be lost), before it accepts my log-off commands.
Were there any instances of data loss or system corruption due to log-off procedures?	When I log off the system, I can be confident that there will be no loss of data or corruption of procedures.
Do control question defaults protect against data loss and data corruption (for example the default for the control question, "Make backup of data?," would be YES)?	The system's "control question defaults" do a good job of preventing data loss or corruption (e.g., the default for the control question, "Make backup of data?," would be YES).
During partial system failure, are data protected when inoperative modes are shut down?	When the system shuts down inoperative modes during a partial system failure, it does a good job of protecting my data.
Were there any instances of data loss or procedural corruption due to accidental actions or commands?	If I initiate accidental actions or commands, the system does a good job of preventing data loss and corruption of system procedures.
Are ordinary control actions simple and direct, while destructive actions are more difficult and require more steps?	The system does a good job of protecting data, because ordinary control actions are simple and direct, while destructive actions are more difficult and require more steps to initiate.
Does the system provide an alarm or warning message for situations involving potential data loss?	The system gives me adequate alarms and warning messages whenever a situation involves potential data loss.
Normal Operating Feedback: These items concern whether the software interface gives adequate feedback to the user during system operations (e.g., PRINTING, WAIT, COPYING, DELETING, STANDBY).	

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.3. Potential Checklist and Questionnaire Items -- Error Management Category (Continued)

Does the system provide feedback including system status information, confirmation, and verification throughout all interactions with the system?	The system gives me effective feedback (including system status information, confirmation, and verification), throughout all my interactions with the system.
Does the system provide messages such as WORKING, BUSY or WAIT during standby?	The system gives me effective, noticeable messages (e.g., WORKING, BUSY or WAIT), when I have to wait for it to finish a processing sequence.
After a process or function is completed or aborted, does the system display the outcome of the process or function?	After the system completes (or aborts) a process or function, it always displays obvious, effective feedback about the outcome of the process or function.
Is system status information on operational modes, availability, and loads available at all times?	Whenever I need it, it is easy to get complete, accurate system status information on operational modes, availability, and system loads.
When the system rejects an input, does it provide feedback showing the reason for the rejection?	When the system rejects an input, it gives me obvious, helpful feedback showing why it rejected the input.
Does the feedback for rejected inputs contain any directions for required corrective actions(s)?	When the system rejects my input, it gives feedback that contains simple, effective directions for corrective action(s).
Do feedback messages use abbreviations only when necessary?	The system appears to use abbreviations in its feedback messages only when necessary.
Does the system provide a warning when a transaction or process invoked will be time consuming or expensive?	The system gives me adequate warning before it begins a processing sequence that will be time consuming or expensive.
Does the system provide informative feedback with all alarms?	Whenever the system gives me an alarm, it also provides effective, informative feedback.

Error Feedback:

These items concern whether the software interface gives adequate feedback to the user to correct and prevent user errors.

Do error messages provide as much diagnostic and remedial material as can be deduced from the error condition?	Error messages always give me all the diagnostic and correction information I need to resume normal use of the system.
Are error messages brief, specific, and task-oriented?	Error messages are always brief, specific and task-oriented.
When the system displays an error message, does it use a cursor to mark the location of the error itself?	When the system displays an error message, it also uses a noticeable cursor to mark the location of the error itself.
Can the errors be displayed or edited sequentially or by priority?	When the system gives me an error message, it is easy to display or edit errors sequentially, or by priority.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.3. Potential Checklist and Questionnaire Items -- Error Management Category (Continued)

Prompts: These items concern the adequacy, location, and clarity of any software interface prompts or messages.	
Does the system prompt the user when missing data are detected?	The system gives me a noticeable, useful prompt when it detects missing or incomplete data.
Does the system display prompts and prompt messages in standard locations?	It is easy to notice system prompts because the system displays prompts and prompt messages in standard locations.
How clear and understandable are the prompts and prompt messages?	System prompts are usually noticeable, clear, and understandable.
Can the system prompts be understood without using references?	It is easy for me to understand and respond to system prompts without using references.
Defaults: These items concern the adequacy and ease of use of any system defaults for data entry.	
Does the system automatically display the currently defined default values when the user initiates a data entry transaction?	Data entry is easy because the system automatically displays the currently defined default values of data entry fields (on special data entry screens).
Can the values of defaults be generated or changed upon request?	It is easy to display or change values of system defaults.
Can all default values in a predefined series be accepted, instead of separately accepting each default in the series?	On menus of system default settings, it is easy to accept the entire series of recommended default values, instead of accepting each default value one-by-one.
Job Aids: These items concern any guidance or help functions provided at the software interface.	
Can guidance information (such as help) be displayed at any step in a transaction sequence?	I can easily display guidance information (such as on-line help) at any step in a sequence of commands.
If the user enters, "HELP" or triggers a help key (or on-screen button), does the system provide help or guidance?	It is easy to get system help or guidance by entering "HELP" or triggering a help key (or on-screen button).
↑ **CHECKLIST ITEMS**	↑ **QUESTIONNAIRE ITEMS**

Table C.3. Potential Checklist and Questionnaire Items -- Error Management Category (Continued)

Is the help provided in response to help requests applicable to task context and the current transaction (i.e., is there context-sensitive help?)?	Whenever I need it, I can easily get "context-sensitive help" that applies to the task or transaction I am doing at the time.
Does the system provide an on-line dictionary of abbreviations and codes?	It is easy to access on-line dictionaries of abbreviations and codes.
After supplying an error message for an error, can the system provide additional guidance upon request?	If the system displays an error message, it is easy to get additional guidance from the system if I want it.
Does the system provide a list of available control options upon request?	If I need a list of available control options, it is easy to get the system to provide it.
Usage Records: These items concern the adequacy and usability of system-generated records of users, transactions, transmissions, data access, and error rates.	
Does the system maintain records of user transactions?	The system maintains adequate records of my transactions with the computer.
Do transaction records include the duration, sequencing, and frequency of the different transactions?	The system maintains adequate records of the duration, sequencing, and frequency of the different transactions between me and the computer.
Does the system maintain records of data access by users?	The system maintains adequate records of my access to data.
Do the records for data access include which data files and items have been accessed?	The system maintains adequate records of the specific data files and items that I access.
Does the system keep records of user errors?	The system maintains adequate records of the errors that I make.
Does the system keep records of system processing errors?	The system maintains adequate records of the errors that it makes.
Does the system automatically log all data transmissions and keep a record of them and their outcomes?	The system maintains adequate records of all my data transmissions, including whether they were received successfully.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.4. Potential Checklist and Questionnaire Items -- Interactive Control Category

Interactive Control:

This category of human-software interactions encompasses the methods by which the software interface allows the user to manipulate, direct, and govern system operation. It includes the dialog type, organization of transactions, varieties of manipulations, system responsiveness, and the ability of the user to control the software interface.

Dialog Type - Question and Answer:

These items concern the software interface's use of questions and answers.

When the system asks a series of related questions, are all previous questions and answers also displayed (to provide context for the current question)?	When the system asks me a series of related questions, it also displays all previous questions and answers (to provide context for the current question).
When the system asks a series of related questions, does the order of the questions match the order of the data from the source document?	When the system asks me a series of related questions about a standard source document, the order of the questions on my screen matches the order of the data from the source document.

Dialog Type - Menu Selection:

These items concern the adequacy and ease of use of menus at the software interface.

Does each menu require one (and only one) selection?	When I select commands from a menu, I never have to select more than one item from the same menu.
Do menus provide only one option per line of text?	When I select commands from a menu, I never see more than one option per line of text in the menu.
Does the selection of any menu option require two actions (e.g., the first to choose the option and the second to start the process or task)?	Selecting a menu option always requires two simple actions (i.e., the first to choose the option and the second to start the process or task).
Does the system provide a standard command entry area (or window) to enter the selected menu option code?	When I select menu items by entering a code or keystroke, the system provides a standard command entry area (or window) to enter the selected menu option code.
Does the system have menu options that are phrased only as commands to the computer (e.g., Fire Rockets, Do Routine, Run BIT/BITE)?	Menu options are phrased only as my commands to the computer (e.g., Fire Rockets, Do Routine, Run BIT/BITE).

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.4. Potential Checklist and Questionnaire Items -- Interactive Control Category (Continued)

Do menus display all options available at the current step in a transaction sequence?	Menus clearly display all options available at the current step in a transaction sequence.
Do menus suppress all options that are not available at the current step in a transaction sequence?	Menus clearly suppress all options that are not available at the current step in a transaction sequence.
Does the system list menu options in either logical order or by their frequency of use?	The system always lists menu options in either logical order or by their frequency of use.
Does the system have menus that are logically related groups of options, not a long list of alternatives?	The system has menus that are logically related groups of options, not merely a long list of alternatives.
Does a top level menu serve as a consistent starting point or "home base" for control entries?	There is always an easy way to reach a top level menu that serves as a consistent starting point or "home base" for control entries.
Do the system menus permit immediate access to critical or frequently selected options?	The system's menus allow me to gain simple, easy, immediate access to critical or frequently selected options.
Does the system indicate the user's current position within the menu sequence structure?	The system gives a highly visible indication of my current position within any menu sequence structure.
Can a command entry be made directly, bypassing a series of menu selections?	I can easily make a command entry directly, bypassing a series of menu selections.
Does each menu have a title that identifies the purpose of that menu?	Each menu has a title that clearly identifies the purpose of that menu.
Does the user select menu options by using the first (or key) letter(s) of their labels, not an arbitrary number?	I can easily select menu options by using the first (or key) letter(s) of their labels, not just an arbitrary number.
Can the previous menu be reached by using a single key or mouse action?	If I need to, I can easily return to the previous menu by using a single key or mouse action.
Can the original (main or top level) menu be reached by using a single key or mouse action?	If I need to, I can easily return to the original (main or top level) menu by using a single key or mouse action.

Dialog Type - Programmable Function Keys:

These items concern the adequacy and ease of use of variable-function or programmable keys (e.g., F-keys). [Other types of keys include fixed single-function keys (e.g., BACKSPACE) and fixed multifunction keys (e.g., upper and lower case alphabet keys).]

Does the system temporarily disable variable-function keys that should not be used for the current task or transaction?	The system helps me avoid errors by temporarily disabling variable-function keys that should not be used for the current task or transaction.
Does the system indicate which variable-function keys are currently active?	The system gives a noticeable indication of the variable-function keys that are currently active.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.4. Potential Checklist and Questionnaire Items -- Interactive Control Category (Continued)

If a function is used in more than one software program, is that function always assigned to the same variable-function key?	It helps me a lot that if a function is used in more than one software program, that function is always assigned to the same variable-function key.
Does the system use variable-function keys in an efficient and productive manner?	The way the system uses variable-function keys helps me to be more efficient and productive.
Dialog Type - Command Language: These items concern the adequacy and ease of use of any command language needed to interact with the system.	
Does the system provide a consistently located command entry area?	It helps me a lot that the system always displays the command entry area in the same place on the screen.
Are commands phrased only as directions to the computer (e.g., Fire Rockets, Locate Target, Run BIT/BITE)?	When I enter commands myself, they are always phrased only as directions to the computer (e.g., Fire Rockets, Locate Target, Run BIT/BITE).
Are commands that are different in function, distinctive from one another?	When I enter commands myself, commands that are different in function are distinctly different from one another.
Can abbreviations be entered for commands longer than five characters?	When I enter commands myself, I can use simple abbreviations for commands that are longer than five characters.
Do simple commands exist for inexperienced users and a complete (more complex) set with all options or manipulations for experienced users?	When I enter commands myself, there is a simple set of commands for inexperienced users and a more complete set (with all options or manipulations) for experienced users.
Dialog Type - Query Language: These items concern the adequacy and ease of use of any query language needed to interact with the system.	
Can alternative queries be used (same request, different wording), like common alternatives in normal language?	When I query a data base, it helps that I can use alternative queries (i.e., same request, different wording), just like I can ask the same question different ways in ordinary English.
Are there any instances in which a query must indicate the storage or structure of data to retrieve data?	When I query a data base, I can ask for the data I want directly. I never have to include the type of data storage or data structure to retrieve data.
Can sequential queries be linked as a single entry by logical elements (e.g., "and", "or", ".not.")?	When I query a data base, I can easily link sequential queries into a single entry using logical elements (e.g., "and", "or", ".not.").

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.4. Potential Checklist and Questionnaire Items -- Interactive Control Category (Continued)

Dialog Type - Graphical Interaction: These items concern the adequacy and ease of use of icons and symbols found at the software interface.	
Do graphical aids supplement other types of interactive control?	When the system uses control icons for me to select functions or programs, I also have the option to use simple nongraphical ways to select the same functions or programs.
Does the system use text labels with control icons (to ensure that the user will understand the icons)?	When the system uses control icons, it also uses legible text labels to ensure that I will understand the icons.
Are the control icons used for system control easy to understand?	When the system uses control icons, the graphical icons are very easy to understand.
Dialog Type - Data Transmission: These items concern the adequacy and ease of use of software interface functions that send and receive data and messages.	
Does the system use consistent data transmission procedures?	It is easy to understand and follow the data transmission procedures in all software programs or functions that allow message and data transmission.
Can data be transmitted from both data displays and stored data files?	It is easy to transmit data from both data displays and stored data files.
Can message and device priorities be assigned for both sending and receiving data transmissions?	It is easy to assign message and device priorities for both sending and receiving data transmissions.
Does the system provide an on-line directory showing all acceptable forms of message addressing?	The system provides an easy-to-use, on-line directory showing all acceptable forms of message addressing.
Does the system provide an on-line directory showing all destinations in the system?	The system provides an easy-to-use, on-line directory showing all destinations in the system.
Does the system provide an on-line-directory showing all links to external systems?	The system provides an easy-to-use, on-line directory showing all links to external systems.
Can addresses be taken from the directory for direct insertion into a header, message, or address specification?	It is easy to take addresses from the on-line directory and insert them directly into a header, message, or address specification.
If transmission fails, can existing messages be re-sent without complete reentry?	If data transmission fails, I can easily re-send my existing messages without completely reentering them from the keyboard.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.4. Potential Checklist and Questionnaire Items -- Interactive Control Category (Continued)

Transaction Selection: These items concern the adequacy and ease of use of software interface functions that allow the user to control current tasks and tasks to be performed in the future.	
Can the sequence of transactions be selected without regard to system processing?	I can easily select the sequence of transactions I want the system to do, without regard to system processing.
Can a single name or code (i.e., a macro) be assigned to a series of commands for use as a command entry?	I can easily assign a single name or code (i.e., a macro) to a series of commands, for use as a command entry.
Can the start time (and completion time, if applicable) be specified when programming the system to do an unattended processing task?	I can easily specify the start time (and completion time, if applicable) when programming the system to do an unattended processing task.
Can the system automatically do programmed periodic processing tasks?	It is easy to set up the system to do periodic tasks automatically.
Do similar, logically related transactions use standard procedures?	The system is easy to use, because similar, logically related transactions use standard procedures.
Interrupt: These items concern whether the user can easily interrupt (and resume) a processing task without losing data.	
Is the user able to CANCEL a processing task (that will erase any changes he has just made and restore the display to the previous condition)?	I can easily CANCEL a processing task (to erase any changes I have just made and restore the display to its previous condition).
Is the user able to interrupt a processing task, and use a nondestructive BACK UP option (to return the display to the previous condition without erasing the latest changes)?	I can easily interrupt a processing task, and use a nondestructive BACK UP option (to return the display to the previous condition without erasing the latest changes).
When the user interrupts a processing task, is a RESTART option available (that causes the system to return to the first position, allowing sequential viewing and editing of all transactions)?	I can easily RESTART a processing task (to cause the system to return to its first position, allowing sequential viewing and editing of all transactions in the task).
Is the user able to ABORT a defined transaction sequence (that will cancel any entries made in the defined transaction sequence and return to the beginning of the sequence)?	I can easily ABORT a processing task (to cancel any entries made in the defined transaction sequence and return to the beginning of the sequence).
When the user ABORTS a defined transaction sequence, does the system require confirmation of the ABORT command before it will begin the ABORT action?	Before the system will ABORT a processing task, it requires me to confirm the ABORT command.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

Table C.4. Potential Checklist and Questionnaire Items -- Interactive Control Category (Continued)

Is the user able to STOP a repetitive (i.e., iterative) processing task without losing any data?	I can easily STOP a repetitive (i.e., iterative) processing task without losing any data.
Is the user able to PAUSE/CONTINUE a processing task (i.e., interrupt and resume a processing task without losing any data)?	I can easily PAUSE/CONTINUE (i.e., interrupt and resume) a processing task without losing any data.
When the user PAUSES a processing task, does the system display an indication that the system is PAUSED?	When I PAUSE a processing task, the system displays a noticeable indication that the system is PAUSED.
Is the user able to SUSPEND a processing task (that allows him to log-off the system and log-on again without affecting data entries or control logic)?	I can easily SUSPEND a processing task (that allows me to log-off the system and log-on again without affecting data entries or control logic).
When the user SUSPENDS a processing task, does the system display an indication that the system is SUSPENDED?	When I SUSPEND a processing task, the system displays a noticeable indication that the system is SUSPENDED.
Control Relationship: These are general items about the experiences and expectations of users.	
Does the design of system interactive controls minimize the need for control actions by the user?	The design of the system's interactive controls makes it simple for me to use a minimum of control actions.
Are controls compatible with the lowest anticipated skill level of the users?	I believe that the system's interactive controls will be compatible with the lowest skill levels of the people who are expected to use this software interface.
Can steps in control actions be skipped or bypassed (as an option) by experienced users?	Experienced users will easily be able to exercise the option to skip or bypass steps in control transactions.
Is the choice of control method(s) compatible with task requirements and user expectations?	I believe the control method(s) used by the system are compatible with both the types of tasks I have in my job and how I expect to do them.
Can a user control the system without extensive knowledge of the internal data storage and retrieval mechanisms used?	I can easily control the system without having extensive knowledge of the internal data storage and retrieval mechanisms that it uses.
Do user personnel control the system, not being controlled or paced by internal processing constraints?	I feel that I am in control of the system, not being controlled or paced by internal processing constraints of the system.

↑ **CHECKLIST ITEMS** ↑ **QUESTIONNAIRE ITEMS**

APPENDIX D
EXAMPLE BIOGRAPHICAL DATA QUESTIONNAIRE

This appendix contains a sample questionnaire of useful biographical data on Test Players. Questions are to be selected from this listing and added to the front of the usability questionnaire or used as a separate questionnaire. Additional questions may be appropriate and should be included as needed.

TEST PLAYER CONTROL NUMBER: ____________

INFORMATION: Although the provisions of the Privacy Act do not apply (because we are not asking for your name or Social Security Number), we want you to know how we plan to use the information you give us on this form. This information will not be released to any agency not having a need to know. Any biographical information released will be only in the form of summarized group statistics.

PRINCIPAL PURPOSE: To summarize the education, experience, and personnel characteristics of [SYSTEM] users/administrators [ETC...] who participated in the [TEST]. The summary will be compared to estimates of the personnel characteristics of the entire population of people who can be expected to operate and use components of the [SYSTEM]. The comparison will allow us to verify the adequacy of the human factors engineering, manpower, personnel, training, system safety, and health hazards characteristics of the contractor's solution to the [SYSTEM] requirement, particularly those characteristics relevant to the operation and use of the system.

INSTRUCTIONS: Please fill out as completely and as legibly as possible (please use a blue or black ball point pen). If you have any questions please ask the data collector.

AGE: _____ years (nearest whole year)

SEX: (Circle one) Male/ Female

MILITARY - ____________________
(Circle Status and Pay Grade)

	STATUS	PAY GRADE				
Commissioned Officer:	O	1	2	3	4	5
Warrant Officer:	W	6	7	8	9	10
Enlisted:	E					

MOS/ASI _____/_____

ADDITIONAL MOS(s) _____ _____ _____

CIVILIAN EMPLOYEES - ____________________

GRADE LEVEL __________

JOB SERIES
(Number/Title)____________________

TIME IN GRADE: ____ years ____ months

TIME IN MOS: ____ years ____ months

MILITARY SERVICE: ____ years ____ months

TIME IN GRADE: ____ years ____ months

TIME IN SERIES: ____ years ____ months

CIVILIAN SERVICE: ____ years ____ months

ASSIGNED UNIT:____________________

YEARS AT PRESENT UNIT:_____years _____months

PRESENT DUTY ASSIGNMENT (i.e., your job):____________________

YEARS AT PRESENT DUTY ASSIGNMENT: _____ years _____ months

DUTY ASSIGNMENT IS: _________ Full Time _________ Part Time

ROLE DURING [TEST] (Check YES or NO to all):

TEST PLAYER ROLE	YES	NO
System Administrator		
User		
Operator		
Security Officer		

HEIGHT: ______inches (nearest inch)   WEIGHT: _____lbs (nearest lb)

DO YOU WEAR CORRECTIVE GLASSES/LENSES . . . (Check YES or NO to all):

USE GLASSES	YES	NO
for Reading?		
for Distance?		
to see a Computer Screen?		

ARE YOU COLOR BLIND? _____ YES   _____ NO

WHAT IS YOUR VISUAL ACUITY (e.g., 20/20), WITH GLASSES IF YOU WEAR THEM? 20 /______

WHICH ONE IS YOUR DOMINANT HAND
(i.e., the one you write with)?: _____ RIGHT   _____ LEFT

CIVILIAN EDUCATION LEVEL: _____ years (nearest whole year)

HIGHEST DEGREE HELD:____________________

OTHER DEGREES/ CERTIFICATES: ____________________

MILITARY EDUCATION: ____________________

____

____

____

WHAT TYPE OF TRAINING, IF ANY, DID YOU RECEIVE ON THIS SYSTEM? ____________________

____

WHEN WAS THIS TRAINING RECEIVED? (Enter the starting date - mm/dd/yy): ___/___/___

HOW LONG DID IT LAST? _____ months _____ weeks _____ days

LIST ANY EXPERIENCE AND OTHER SCHOOLING IN AUTOMATION THAT YOU HAVE RECEIVED:

____

OF THE FOLLOWING DEVICES, SOFTWARE, AND SYSTEMS, CHECK THOSE THAT YOU HAVE PERSONALLY USED AND ARE FAMILIAR WITH. (Check YES or No for all)

YES	NO		YES	NO	
____	____	Keyboard	____	____	Numeric key pad
____	____	Mouse	____	____	Light pen
____	____	Touch screen	____	____	Trackball
____	____	Joystick	____	____	Text editor
____	____	Unix	____	____	Windows
____	____	Graphical User Interface	____	____	MS-DOS
____	____	OS/2	____	____	Word processor
____	____	File manager	____	____	Spreadsheet
____	____	Electronic mail	____	____	Database programs
____	____	Desktop publishing	____	____	Multi-media
____	____	CAD/CAM	____	____	Graphics Software
____	____	Software utilities/tools	____	____	Video arcade games
____	____	Computer monitor/video display	____	____	Computer assisted instruction (CAI)
____	____	Mainframe	____	____	Personal computer
____	____	Laptop/Notebook computer	____	____	Computer magazines
____	____	Computer users' group	____	____	Floppy disks
____	____	Hard disks			

WHAT WAS YOUR FREQUENCY OF COMPUTER USAGE PRIOR TO THE START OF TEST? (Please check only one period of usage, and enter the approximate hours that you used the computer during the period you have selected.):

FREQUENCY OF COMPUTER USAGE PRIOR TO BECOMING A TEST PLAYER	CHECK (One Only)	HOURS OF USAGE PER PERIOD
Daily		
Weekly		
Monthly		
Quarterly		
Semi-Annually		
Annually		
NEVER		0

HOW MANY DIFFERENT TYPES OF COMPUTER SYSTEMS HAD YOU WORKED WITH PRIOR TO BECOMING A TEST PLAYER IN THIS TEST?

(Enter 0 or a whole digit): ____

APPENDIX E
EXAMPLE END-OF-TASK QUESTIONNAIRE

END-OF-TASK QUESTIONNAIRE -- SYSTEM LUT

______________ **Software Interface**

[OPTIONAL: Insert the definition of the software interface in terms of Programs, User Roles, and User Functions.]

Test Player Control Number (TPCN): __________

Test Event Number: __________

Task Number: __________

Task Starting Time (24-hr clock) (hh:mm): _____:_____

Day of the Week: __________

Date (dd/mm/yy): ___/___/___

When did you stop working on the task? Time: (hh:mm): _____:_____

Did you complete the task? _____ Yes _____ No

INSTRUCTIONS:

Answer by writing in the space provided, by checking Yes or No, or by selecting the nearest whole number that best describes how much you agree or disagree with each statement. Please fill out this questionnaire as completely and as legibly as you can. Use a blue or black ballpoint pen. If you have any questions please ask a data collector. If you need additional space for comments, use the last page of the questionnaire. Be sure to start each comment with the Item Number (e.g., #3).

Answer all rating scale statements using the scale below. Write your answer to each statement in the blank preceding the statement.

DISAGREE COMPLETELY				NOT SURE DON'T KNOW				AGREE COMPLETELY
1	2	3	4	5 ?	6	7	8	9

(Select the appropriate whole number)

1. _____ The system does all that it should to help me do this task successfully.

TPCM: ____________

DISAGREE COMPLETELY				NOT SURE DON'T KNOW				AGREE COMPLETELY
1	2	3	4	5 ?	6	7	8	9

(Select the appropriate whole number)

2. _____ The system response time was adequate during this task.

3. _____ The training prepared me to use all the capabilities of the software interface that I needed to do this task.

4. _____ I did not have to discover (or be given) new ways to use the software interface to do this task.

5. _____ The data on a single screen gave me all the information I needed to do this task.

6. _____ I did not have to combine data from different screens or printouts to do this task.

7. _____ The on-line Help functions alone were sufficient to help me do this task.

8. _____ Access to on-line Help functions for this task was quick and easy.

9. _____ The "Help Desk" gave me adequate assistance to do this task.

10. _____ Recovery from, and correction of, input errors was easy.

11. _____ It was easy to execute commands.

12. _____ I had no trouble getting my output to print.

13. _____ I never had to input the same data more than once.

14. _____ The functions of numeric, function, and control keys were the same as for other tasks.

15. _____ The screen was easy to read and use while I performed my task.

16. _____ Security procedures were not a problem during this task.

17. _____ The right <u>numbers</u> of users (including me) were located at my work site to do/assist with, this task.

18. _____ The right <u>type</u> of users (by MOS/ASI/Grade, including me) were located at my work site to do/assist with, this task.

19.

19. While doing this task, I developed:	YES	NO
a. A headache.		
b. Arm, or wrist soreness.		
c. Neck, or back fatigue.		
d. Eyestrain.		

TPCN: ____________

20.

20. While doing this task, I had trouble reading the following on the screen:	YES	NO
a. Text or letters.		
b. Numbers.		
c. Symbols or icons.		

21. Did you notice any features or operating characteristics of any part of the system that you think could be improved? (Report only suggestions that you have not made before.)

a. **Hardware?** (Controls, displays, other):

b. **Software?** (Applications, interfaces, other):

c. **System Procedures?** (Users' procedures that are not programmed or hard-wired into the system):

22. Did you notice any feature of the system **that could pose a potential:** (Report only conditions that you have not reported before.)

a. Safety or health **hazard to people?**

TPCN: ___________

b. Safety **hazard to equipment?**

c. Security or integrity **risk to data or software?**

ADDITIONAL COMMENTS:

APPENDIX F
EXAMPLE USABILITY PROFILE QUESTIONNAIRE

POST-TEST QUESTIONNAIRE -- SYSTEM LUT

________________ Software Interface

[OPTIONAL: Insert the definition of the software interface in terms of Programs, User Roles, and User Functions.]

DATE (dd/mm/yy): ___/___/___

Test Player Control Number (TPCN): __________

INSTRUCTIONS:

We would like to have your opinions of various features of the ______________ Software Interface of the (system). Please fill out this questionnaire as completely and as legibly as you can. Use a blue or black ballpoint pen. If you have any questions please ask a data collector. If you need additional space for comments, use the last page of the questionnaire. Be sure to start each comment with the Part/Item Number (e.g., Part 1, #3).

PART I -- TRAINING

Please take a few minutes to tell us your opinions of the ______________ Software Interface part of the training course now that you have had a chance to use the software interface more extensively. Circle the Y or the N before each training item to answer the question.

1. Y N Are there any tasks or system functions that should be added to the system training for this interface?

   COMMENTS: ____

   ____

   ____

   ____

   ____

2. Y N Did you receive any training for this interface that you believe was unnecessary?

   COMMENTS: ____

   ____

   ____

   ____

   ____

TPCN: ___________

ADDITIONAL INSTRUCTIONS: Beginning with Item 3 below, through the remainder of Parts II and III, answer by selecting the nearest whole number that best describes how much you agree or disagree with each statement. If you need to make comments, use the last page of the questionnaire. Be sure to start each comment with the Part/Item Number (e.g., Part 1, #3).

Answer all remaining questionnaire statements using the rating scale below. Write your answer to each statement in the blank preceding the statement.

DISAGREE COMPLETELY				NOT SURE DON'T KNOW				**AGREE** COMPLETELY
1	**2**	**3**	**4**	**5** ?	**6**	**7**	**8**	**9**

(Select the appropriate whole number)

3. ____ The training adequately prepared me to use the __________ Software Interface to do my job.

PART II -- OVERALL RATING

Please answer the following question from the viewpoint of your present job or duty position.

1. ____ The __________ Software Interface works very well.
2. ____ The __________ Software Interface will make my job easier and more productive.

PART III -- USABILITY OF THE INTERFACE

1. ____ Display screens have all the information I need to do my work.
2. ____ Display screens are not cluttered by unnecessary information.
3. ____ I have no trouble finding information on data displays.
4. ____ It is easy to read data and text from data displays.
5. ____ Data shown on the display screen are always in the format I need.
6. ____ I do not have to hand-copy data from the display screen to paper.
7. ____ Even when the data sheet is larger than the display window, it is still easy to find the data I need.
8. ____ I can easily get a printed copy of the screen when I need it.
9. ____ It is easy to change the way screen features (e.g., windows, icons, etc.), are displayed.
10. ____ On-screen written instructions, prompts, and menu selections are easy to understand.

TPCN: ____________

DISAGREE COMPLETELY				NOT SURE DON'T KNOW				AGREE COMPLETELY
1	2	3	4	5 ?	6	7	8	9

(Select the appropriate whole number)

11. _____ The abbreviations, acronyms, and codes used in the display screens rarely cause me to make mistakes.

12. _____ It is always easy to tell what each icon stands for.

13. _____ [Color Monitors:] The displays are easy to read because of contrasting colors.

14. _____ [Monochrome Monitors:] The displays are easy to read because of contrasting shades of gray.

15. _____ It is easy to tell where the cursor is on the screen.

16. _____ I can easily use the mouse to "click"/"double click" on display control features, icons, and menu lines, without making errors.

17. _____ The software interface directs my attention to displayed data that are critical or abnormal.

18. _____ The software interface helps me avoid mistakes when I enter or edit text or graphics, by showing me the attributes I have selected (e.g., typeover/insert, CAPS LOCK, line draw, polygon draw, text entry, etc.).

19. _____ The capability to re-size displays and windows is very useful.

20. _____ Audible signals (e.g., "beeps") help me avoid and correct mistakes.

21. _____ I rarely make accidental keystrokes that cause me to erase data or cancel a command.

22. _____ It is easy to distinguish data entry fields from other parts of the screen where the software interface will not permit me to enter data.

23. _____ Except for special-access or log-on passwords, the display screen always shows me what data I am keyboarding into the system.

24. _____ The software interface does not have confusing requirements to use different data entry procedures from one screen (or application) to the next.

25. _____ I always know approximately how long I will have to wait for the system to finish processing.

26. _____ I rarely have to reenter data that I know is already available to the software interface in other files.

27. _____ The cursor does not prevent me from reading or seeing the data, characters, symbols, or display features that are behind the cursor.

28. _____ It is easy to adjust the cursor's response, movement speed, and accuracy, to my liking.

29. _____ The cursor always moves as I expect it to, from the way I manipulate the cursor controls (keyboard or mouse).

30. _____ When a keystroke (or mouse click) does not immediately produce the response I expect, the software interface always gives me a message, symbol, or sign that acknowledges my keystroke (or mouse click).

TPCN: ____________

DISAGREE COMPLETELY				NOT SURE DON'T KNOW				AGREE COMPLETELY
1	2	3	4	5 ?	6	7	8	9

(Select the appropriate whole number)

31. ____ Fixed-function keys perform the same way in all windows and applications. (Examples: "The BACKSPACE key always deletes the previous character as it moves one space to the left"; or," Except on page 1 of a document, the PAGE UP key always moves the cursor to the previous page, not just to the top of the same page.")

32. ____ When I use scroll functions, it is easy to make the text or document move in the direction I want it to move.

33. ____ It is easy to make the cursor move to the top or bottom (or beginning or end) of a document or file.

34. ____ It is easy to use editing commands (such as move, copy, delete, etc.) to edit written documents, data entry fields, or graphical displays.

35. ____ It is easy to highlight, exactly, the portions of text that I want to move, copy, underline, italicize, delete, etc.

36. ____ It is easy to have the system search forward (and backward) for a string of text (words, phrases, or numbers) that I want to find in a document.

37. ____ In applications that allow me to choose my own format, it is easy to set the format controls for documents (e.g., margins, tabs, line spacing, etc.).

38. ____ In applications that allow me to choose my own format, it is easy to save and recall format control settings for use with other documents.

39. ____ I always have the option to display text exactly as it will be printed (i.e., a print preview, with underlining, boldface, subscript, type styles and sizes, etc.), before I print it.

40 ____ It is easy for me to select and print only one page (or a few pages) from a multi-page document.

41. ____ Accidental keystrokes never cause me to erase or overwrite files or large segments of data or text by mistake.

42. ____ If I make a small data entry or typing error, it is easy for me to correct the error without having to retype the entire entry.

43. ____ Whenever I am about to enter a critical change or take some important, unrecoverable action, the software interface makes me confirm the entry before accepting it.

44. ____ Software interface messages (e.g., error messages; prompts, cues, or unasked-for help; or on-line Help messages) never insult me or hurt my feelings.

45. ____ Software interface messages "talk" to me just as a person trying to do a job, not some kind of computer expert.

46. ____ Error messages always tell me what I have to do to correct the problem.

47. ____ If an error message does not tell me exactly what is wrong, it at least gives me some suggestions or a trouble-shooting process to follow.

48. ____ Using the on-line Help functions in the software interface teaches me how to avoid doing things that cause error messages.

TPCN: ____________

DISAGREE COMPLETELY				NOT SURE DON'T KNOW				AGREE COMPLETELY
1	2	3	4	5 ?	6	7	8	9

(Select the appropriate whole number)

49. _____ I use the same keystrokes or mouse selections to reach the on-line Help functions in all applications.

50. _____ I can browse through on-line Help screens (including "Documents"), similarly to browsing a printed manual.

51. _____ As far as I know, I cannot read files or data to which I have not been granted access.

52. _____ There are no software problems that cause me to lose data.

53. _____ There are no equipment problems that cause me to lose data.

54. _____ System log-on procedures are never an unjustifiable pain in the neck.

55. _____ System log-off procedures prevent me from accidentally losing data and text that I intend to keep.

56. _____ I often forget to log off the system when I step away from my terminal.

57. _____ As far as I know, when I do not grant editing privileges to other users, they are unable to edit my files.

58. _____ When I make a selection with the mouse, or make an entry from the keyboard, I get an immediate response from the system.
(A response is either an error message or some indication that the system has accepted the entry or selection.)

59. _____ I can always tell whether the system is in a processing sequence and not just waiting for me to do something.

60 _____ I never have to worry about how many other users are working on the system while I am.

61. _____ When the system is doing a long processing sequence (one minute and longer), the display lets me see the number of minutes remaining (or the percent of the process that is completed).

62. _____ If the system rejects one of my data inputs, it always gives me a useful feedback message (i.e., tells me why, and what corrective action to take).

63. _____ System feedback messages are self-explanatory.

64. _____ When the system gives me a message, I never have to memorize anything or refer to written documentation.

65. _____ I seldom have to use printed reference material to supplement on-line guidance and instructions.

66. _____ I can use both the mouse and the keyboard to make any menu selection.

67. _____ It is easy to select a menu option.

68. _____ Menu options are consistent in their wording, order, and location.

69. _____ Hierarchical, cascading menus are easy to use.

70 _____ If I "get lost" using an application, it is easy to find the main menu.

TPCN: ____________

DISAGREE COMPLETELY				NOT SURE DON'T KNOW				**AGREE** COMPLETELY
1	2	3	4	5	6	7	8	9

?

(Select the appropriate whole number)

71. ____ It is easy to tell whether a menu option is disabled or currently available.

72. ____ I can backtrack to the previous menu by using a single keystroke or mouse click.

73. ____ The display screen shows me what key function is currently selected for all keys (including F1 through F12) that toggle between functions, or lock functions in or out (unless the selected function is shown by lights on the keyboard or key itself).

74. ____ Data base queries are simple and easy.

75. ____ I can find data in a data base merely by specifying the data I want (i.e., without having to tell the system how to find the data).

76. ____ The procedures for data transmission are the same, even in different applications.

77. ____ It is easy for me to tell what data or files I am actually transmitting.

78. ____ I have no difficulty preparing and editing outgoing messages.

79. ____ I have no difficulty retrieving and reading incoming messages.

80. ____ It is easy to tell whether the messages I send are actually received.

81. ____ As far as I know, I always receive all the messages that are sent to me.

82. ____ I can easily attach messages to existing files before I transmit them.

83. ____ The on-line address directory makes it easy for me to prepare addresses for messages.

84. ____ The software interface lets me know when I receive a high-priority message.

85. ____ I can scan my incoming messages to prioritize them before I "open my mail."

86. ____ It is easy to acknowledge system alarms, signals, and messages.

87. ____ The software interface keeps me informed about what it is doing (so I do not have to "take mental notes").

88. ____ Using the software interface to do my job lets me get much more work done.

89. ____ If I did not use the software interface to do my job, I could not meet my suspense dates and deadlines.

90. ____ I can easily create, edit, and manage my data files and data directories without knowing much about the internal storage and retrieval mechanisms the software interface uses.

ADDITIONAL COMMENTS:

____

APPENDIX G
CHECKLIST AND QUESTIONNAIRE DATA DISPLAYS

G.1 **GENERAL.** This appendix shows how data from the different types of checklists and questionnaires can be reduced and displayed as meaningful results. In each case, data are initially reduced separately for each of the system's software interfaces.

G.2 **CHECKLIST DATA.** Table G.1 shows how checklist data can be displayed in tabular form. The accompanying paragraphs explain the necessary calculations.

Table G.1 Sample Display of Usability Checklist Data

Percent User Friendly Responses and Number of Checklist Items (by Software Interface and Interaction Category)

SOFTWARE INTERFACE	DATA DISPLAY		DATA ENTRY		ERROR MANAGEMENT		INTERACTIVE CONTROL		INTERFACE TOTAL	
	%	#	%	#	%	#	%	#	%	#
SYSTEM ADMINISTRATOR	80%	20	80%	15	55%	20	80%	25	74%	80
PERSONNEL CLERK	76%	25	64%	25	52%	25	60%	25	63%	100
MAINTENANCE CLERK	80%	15	87%	15	90%	20	85%	20	86%	70

G.2.1 <u>Checklist (Single Respondent)</u>. Follow these steps to produce a system table that compares the rates of favorable responses for all the software interfaces of the system. These steps apply when a single checklist has been completed for each interface. (It is not necessary that the same person complete all the checklists.)

1 - Calculate the percentage of all applicable questions that received favorable responses toward the usability of the software interface.

2 - Report the percentage and the total number of checklist items, by interaction category and in the aggregate.

3 - Display the results in a tabular or graphic form that allows one to compare the checklist results for all of the system's software interfaces.

G.2.2 Checklist (Multiple Respondents). When several people fill out a checklist about the same software interface, follow these preliminary steps for that checklist, to produce the same type of results as in Table G.1.

1 - For each applicable question, score a favorable response as a "1" and an unfavorable response as a "0".

2 - Calculate the mean response to each item.

3 - Re-score items as receiving a favorable response if the item mean is 0.5 or higher.

4 - Re-score items as receiving an unfavorable response if the item mean is below 0.5.

5 - Do the remainder of the calculations as if there were a single respondent.

G.3 QUESTIONNAIRE DATA.

G.3.1 Scaled Response Items (End-of-Task Questionnaires). These questionnaires consist of ease-of-use, usability, or task difficulty statements, about a task that the respondent has just completed. Appendix E is an example of an End-of-Task Questionnaire. Data from the scaled response items can be used to calculate the mean rating of each usability statement across all types of tasks. It can also be used to calculate the mean rating of each type of task across all ease-of-use statements. The next two paragraphs show how to use these data to find the usability characteristics of an interface, and the tasks done at that interface, that are in greatest need of improvement by software developers.

G.3.1.1 Rank Usability Statements Across All Tasks. Table G.2 shows how to display a ranked list, from highest to lowest usability, of the more variable usability characteristics of a software interface. The steps to produce this list follow the table.

Table G.2 Sample Display of Usability Statements (End-of-Task)

Usability Statements -- End-of-Task Questionnaire
(Ranked by Decreasing Degree of Usability)

RANK	USABILITY STATEMENT (END-OF-TASK QUESTIONNAIRE)	MEAN RATING	N	STD DEV	RR
1	3. The data on a single screen gave me all the information I needed to do this task.	7.4	26	.3	87%
2	11. It was easy to execute commands.	6.9	27	.5	90%
3	15. The screen was easy to read and use while I performed my task.	6.9	24	.9	80%
15	12. I had no trouble getting my output to print.	4.8	27	.8	90%
16	1. The system does all that it should to help me do this task successfully.	3.4	26	.5	87%
17	6. I did not have to combine data from different screens or printouts to do this task.	3.2	25	.4	83%

1 - For each scaled response item (i.e., usability statement), calculate the mean, sample standard deviation, sample size, and response rate (across all tasks done at a single software interface).

(The item sample size is based only on those people who responded to the item. The item response rate is the percentage of people who completed the questionnaire and responded to the questionnaire item.

2 - Rank all of the usability statements by mean (D),[1] sample size (D),[1] sample standard deviation (A),[2] and response rate (D)[1].

(The variables other than the mean are included in case some task difficulty item means, sample sizes, etc., are tied.)

3 - If the list of questionnaire items (usability statements), is relatively short, display the results for all items.

1 Descending order.

2 Ascending order.

4 - This will provide a ranked list, from highest to lowest usability, of the interactions done at a software interface.

5 - If the list of items is unmanageably long, display the most and least usable software usability characteristics (i.e., the top and bottom thirds or quarters of the ranked list of items).

G.3.1.2 Rank Standardized Tasks Across All Usability Statements. If real users or Test Players are given standardized or scripted tasks to do during a data collection event, you can use the numerical response data from the End-of-Task Questionnaire to find which tasks are the most and least difficult to do using the software interface. Table G.3 shows how to display a ranked list, from lowest to highest difficulty (i.e., highest usability ratings at the top of the list), of the types of tasks done at a software interface. This list shows you not only which tasks are the most inherently difficult or easy, but the tasks for which the software interface does the most or the least to help the user do his job. The steps to produce this list follow the table.

Table G.3 Sample Display of Standardized or Scripted Tasks

Scripted Tasks -- End-of-Task Questionnaire
(Ranked by Increasing Degree of Difficulty)

RANK	TYPE OF TASK (END-OF-TASK QUESTIONNAIRE)	MEAN RATING	N	STD DEV	RR
1	Script 1 -- Send an E-Mail Message	8.4	55	.2	98%
2	Script 6 -- Write & Simple Memorandum	6.2	52	.5	95%
3	Script 2 -- Attach a File to an Outgoing E-Mail Message	5.8	53	.8	94%
4	Script 4 -- Erase Unneeded Documents	4.5	49	1.0	100%
5	Script 3 -- Reformat the Hard Drive	3.8	29	.5	92%
6	Script 5 -- Program a Data Base Query	2.1	36	.3	81%

1 - For each comparable task, calculate the mean, sample standard deviation, sample size, and response rate (across all usability statements answered about a single software interface).

2 - If the list of tasks is relatively short, display the results for all tasks.

3 - This will provide a ranked list, from lowest to highest degree of difficulty, of the standardized tasks done at the interface.

4 - If the list of tasks is unmanageably long, display the easiest and most difficult tasks (i.e., the top and bottom thirds or quarters of the ranked list of tasks).

G.3.2 Scaled Response Items (Post-Test Questionnaires). Appendix F is an example of a Post-Test Questionnaire. The items are usability statements about human-software interactions that apply to a single software interface. The statements are categorized by the type of interaction (i.e., Data Display, Data Entry, Error Management, and Interactive Control). You will construct the Software Usability Profile for your system from these questionnaire items (i.e., software usability items).

G.3.2.1 Rank Usability Statements for Each Software Interface. Table G.4 shows how to display a ranked list, from highest to lowest usability, of the less variable usability characteristics of a software interface. The steps to produce this list follow the table.

Table G.4 Sample Display of Usability Statements (Post-Test)

Usability Statements -- Post-Test Questionnaire
(Ranked by Decreasing Degree of Usability)

RANK	USABILITY STATEMENT (POST-TEST QUESTIONNAIRE)	MEAN RATING	N	STD DEV	RR
1	8. I can easily get a printed copy of the screen when I need it.	8.8	37	.3	100%
2	21. I rarely make accidental keystrokes that cause me to erase data or cancel a command.	8.2	35	.5	95%
3	75. I can find data in a data base merely by specifying the data I want (i.e., without having to tell the system how to find the data).	7.9	36	.8	97%
88	19. The capability to re-size displays and windows is very useful.	5.1	29	.8	78%
89	7. Even when the data sheet is larger than the display window, it is still easy to find the data I need.	4.5	33	.6	89%
90	55. System log-off procedures prevent me from accidentally losing data and text that I intend to keep.	4.0	34	.4	92%

1 - For each usability statement, calculate the mean, sample standard deviation, sample size, and response rate.

(The item sample size is based only on those people who responded to the item. The item response rate is the percentage of people who completed the questionnaire who responded to the questionnaire item.

2 - Rank all of the usability statements by mean (D),[1] sample size (D),[1] sample standard deviation (A),[2] and response rate (D)[1].

(The variables other than the mean are included in case some software usability item means, sample sizes, etc., are tied.)

3 - If the list of usability statements is relatively short, display the results for all items.

4 - This will provide a ranked list, from highest to lowest usability, of the usability characteristics of a software interface.

5 - If the list of items is unmanageably long, display the most and least usable software interface usability characteristics (i.e., the top and bottom thirds or quarters of the ranked list of items).

G.3.2.2 <u>Construct a Software Usability Profile for the System</u>. Table G.5 shows a tabular Usability Profile for a system. The table compares the usability characteristics across all the software interfaces of the system (by interaction category and the aggregate). For the evaluation or assessment report, analysts can calculate the 90% confidence interval for each mean rating component, and display the results in a bar chart like the one shown in Figure G.1.

[1] Descending order.

[2] Ascending order.

Table G.5 Sample Test Report Display

Software Usability Profile for the XYZ System
Software Interface v. Interaction Category[1]

SOFTWARE INTERFACE		DATA DISPLAY	DATA ENTRY	ERROR MANAGEMENT	INTERACTIVE CONTROL	INTERFACE TOTAL
SYSTEM ADMINISTRATOR	Mean	6.17	4.27	5.79	3.96	5.03
	SD	0.46	1.28	1.11	0.87	1.37
	N	59	77	95	77	308
	RR	98%	96%	95%	96%	96%
PERSONNEL CLERK	Mean	7.52	6.91	7.92	6.52	7.24
	SD	0.71	1.03	0.71	1.12	1.06
	N	325	414	404	320	1463
	RR	96%	97%	95%	94%	96%
MAINTENANCE CLERK	Mean	5.66	4.53	3.53	4.08	4.42
	SD	1.55	1.20	0.93	0.78	1.38
	N	106	129	126	105	466
	RR	95%	97%	78%	88%	89%

[1] Software usability ratings are on a scale of 1-9. Higher scores show higher levels of software usability. The theoretical neutral point is 5.0. (Mean = arithmetic mean, SD = sample standard deviation, N = sample size or number of responses, RR = response rate or percentage of all respondents who answered the items.)

1 - For all the usability statements in an interaction category, calculate the mean, sample standard deviation, sample size, and response rate. Calculate the aggregate statistics across all four human-software interaction categories.

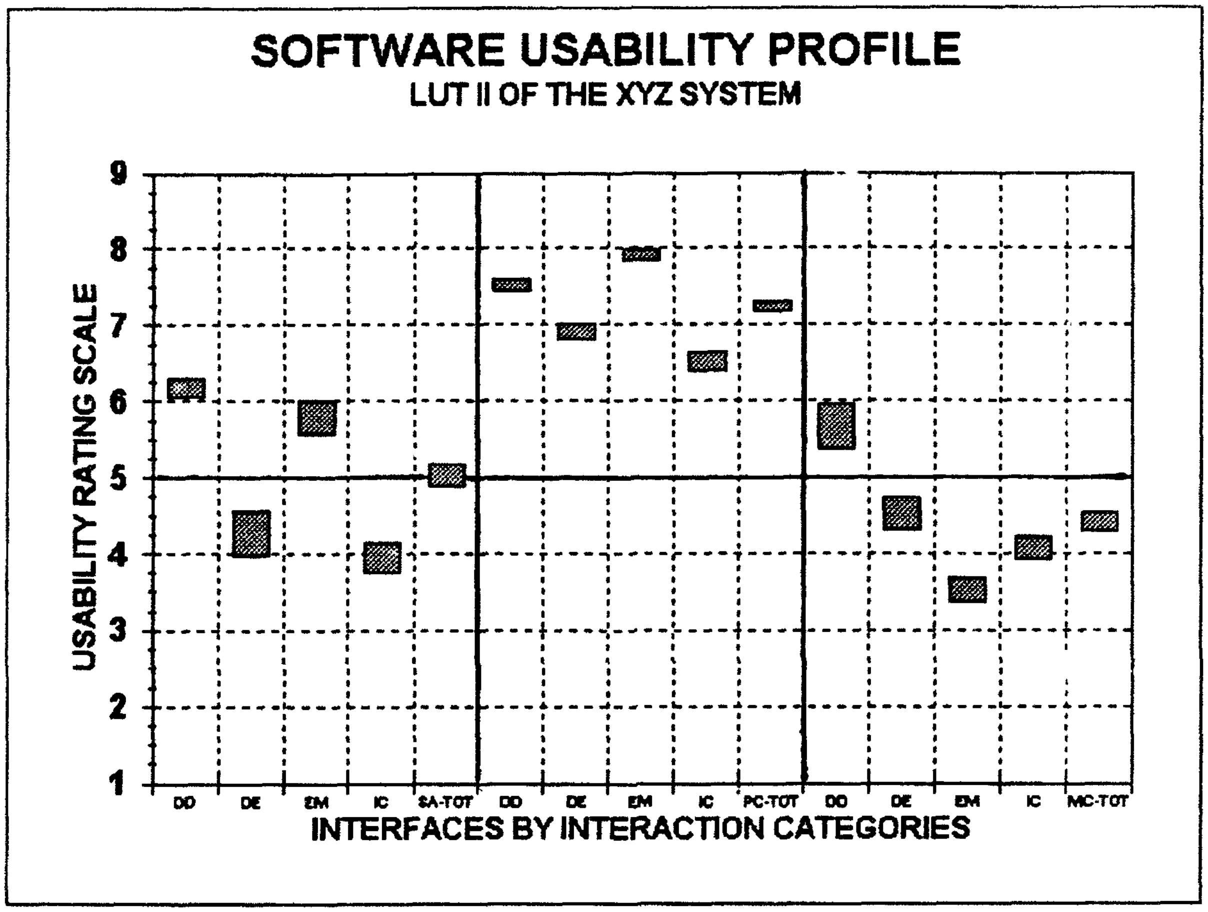
2 - Repeat this for the data from each software interface.

3 - Display the numerical results in the Test Report as shown in Table G.5.

4 - Display the numerical results in the Evaluation or Assessment Report as shown in Figure G.1.

Figure G.1 Sample Evaluation or Assessment Report Display

Software Usability Profile for the XYZ System
90% Confidence Intervals
Interfaces by Interaction Categories v. Usability Rating Scale[1]



[1] Software usability ratings are on a scale of 1-9. Higher scores show higher levels of software usability. The theoretical neutral point is 5.0. (DD = Data Display, DE = Data Entry, EM = Error Management, IC = Interactive Control, TOT = Total Usability Ratings for the Software Interface, SA = System Administrator Interface (on the left, above), PC = Personnel Clerk Interface (in the center, above), MC = Maintenance Clerk (on the right, above))

G.3.3 <u>Other Response Formats</u>.

G.3.3.1 <u>Categorical Response Items</u>. These kinds of questions require structured, "either/or" responses. Examples are YES/NO, TRUE/FALSE, or 1-5/6-10/11-15/16+. Table G.6 shows a small table that compares the response frequencies for such a question.

Table G.6 Sample Display of Categorical Response Item Results

Test Player Reports on Wearing Glasses

Do you wear eyeglasses or lenses?	FREQUENCY	SAMPLE SIZE	PERCENT
YES	21	36	58%
For Reading	10	21	48%
For Distance	11	21	52%
To see a VDT	9	21	43%
All three	8	21	38%
NO	15	36	42%
NO RESPONSE	1	37	3%

1 - Calculate the response frequency for each type of response to the question (e.g., YES, NO, and NO ANSWER).

2 - Report the responses, the response frequencies, the response percentages, and the sample size for each question.

(Calculate the response percentages by dividing each response frequency by the sample size of the question, which is normally the sum of the response frequencies.)

G.3.3.2 Unformatted Response Items (Short Comments). Follow these steps to summarize the short comments that questionnaire respondents write after some formatted response items. The result is an outline of paraphrased responses, including frequencies, subtotals, and totals. Where possible, display the results right after the questionnaire item about which the comments were made. Table G.7 is an example of how the frequencies of short comments can be displayed.

Table G.7 Sample Display of Unformatted Short Comments

- 10 - Data Display
  - 5 - Screen Features/Characters too Small
    - 2 - Menu boxes too small
    - 2 - Numbers difficult to read because of size
    - 1 - Couldn't get mouse cursor on small button
  - 5 - Screen colors were easy to change
- 8 - Interactive Control
  - 5 - Lack of Prompts/Feedback to the User
    - 3 - Screen gave no feedback that system was trying to print
    - 2 - Double-clicked twice on icon because I couldn't tell if the system was responding
  - 3 - The buzzer didn't always sound when I made an error
- 7 - Human Performance and Training
  - 4 - Training was too short
  - 2 - Training did not cover context sensitive help
  - 1 - It was so easy a child could do it
- 4 - Human Factors Engineering
  - 4 - Display flickered

NOTE: Underlined topics are provided by the analyst. Remaining comments are summaries of Test Player comments.

1 - Cluster, summarize, and paraphrase the comments into as few meaningful categories as possible.

2 - Show the response frequencies for each category.

3 - If the structure of the categories also fits a meaningful pattern, you may name larger categories and show the frequency subtotals of all the responses under the larger categories.

(Use some sort of notation such as underlining or upper case to identify categories provided by the analyst.)

4 - The result is an outline of paraphrased responses, with the addition of frequencies, subtotals, and totals.