GP19-A2
Vol. 23 No. 4
Replaces GP19-A
Vol. 15 No. 23

Laboratory Instruments and Data Management Systems: Design of Software User Interfaces and End-User Software Systems Validation, Operation, and Monitoring; Approved Guideline—Second Edition

CLINICAL AND LABORATORY STANDARDS INSTITUTE™

(Formerly NCCLS)
Providing NCCLS standards and guidelines, ISO/TC 212 standards, and ISO/TC 76 standards

This document identifies important factors that designers and laboratory managers should consider when developing new software-driven systems and selecting software user interfaces. Also included are simple rules to help prepare validation protocols for assessing the functionality and dependability of software.

A guideline for global application developed through the NCCLS consensus process.

NCCLS

NCCLS...

Serving the World's Medical Science Community Through Voluntary Consensus

NCCLS is an international, interdisciplinary, nonprofit, standards-developing, and educational organization that promotes the development and use of voluntary consensus standards and guidelines within the healthcare community. It is recognized worldwide for the application of its unique consensus process in the development of standards and guidelines for patient testing and related healthcare issues. NCCLS is based on the principle that consensus is an effective and cost-effective way to improve patient testing and healthcare services.

In addition to developing and promoting the use of voluntary consensus standards and guidelines, NCCLS provides an open and unbiased forum to address critical issues affecting the quality of patient testing and health care.

PUBLICATIONS

An NCCLS document is published as a standard, guideline, or committee report.

Standard A document developed through the consensus process that clearly identifies specific, essential requirements for materials, methods, or practices for use in an unmodified form. A standard may, in addition, contain discretionary elements, which are clearly identified.

Guideline A document developed through the consensus process describing criteria for a general operating practice, procedure, or material for voluntary use. A guideline may be used as written or modified by the user to fit specific needs.

Report A document that has not been subjected to consensus review and is released by the Board of Directors.

CONSENSUS PROCESS

The NCCLS voluntary consensus process is a protocol establishing formal criteria for:

- the authorization of a project
- the development and open review of documents
- the revision of documents in response to comments by users
- the acceptance of a document as a consensus standard or guideline.

Most NCCLS documents are subject to two levels of consensus—"proposed" and "approved." Depending on the need for field evaluation or data collection, documents may also be made available for review at an intermediate (i.e., "tentative") consensus level.

Proposed An NCCLS consensus document undergoes the first stage of review by the healthcare community as a proposed standard or guideline. The document should receive a wide and thorough technical review, including an overall review of its scope, approach, and utility, and a line-by-line review of its technical and editorial content.

Tentative A tentative standard or guideline is made available for review and comment only when a recommended method has a well-defined need for a field evaluation or when a recommended protocol requires that specific data be collected. It should be reviewed to ensure its utility.

Approved An approved standard or guideline has achieved consensus within the healthcare community. It should be reviewed to assess the utility of the final document, to ensure attainment of consensus (i.e., that comments on earlier versions have been satisfactorily addressed), and to identify the need for additional consensus documents.

NCCLS standards and guidelines represent a consensus opinion on good practices and reflect the substantial agreement by materially affected, competent, and interested parties obtained by following NCCLS's established consensus procedures. Provisions in NCCLS standards and guidelines may be more or less stringent than applicable regulations. Consequently, conformance to this voluntary consensus document does not relieve the user of responsibility for compliance with applicable regulations.

COMMENTS

The comments of users are essential to the consensus process. Anyone may submit a comment, and all comments are addressed, according to the consensus process, by the NCCLS committee that wrote the document. All comments, including those that result in a change to the document when published at the next consensus level and those that do not result in a change, are responded to by the committee in an appendix to the document. Readers are strongly encouraged to comment in any form and at any time on any NCCLS document. Address comments to the NCCLS Executive Offices, 940 West Valley Road, Suite 1400, Wayne, PA 19087, USA.

VOLUNTEER PARTICIPATION

Healthcare professionals in all specialties are urged to volunteer for participation in NCCLS projects. Please contact the NCCLS Executive Offices for additional information on committee participation.

GP19-A2
ISBN 1-56238-484-8
Volume 23 Number 4
ISSN 0273-3099

Laboratory Instruments and Data Management Systems: Design of Software User Interfaces and End-User Software Systems Validation, Operation, and Monitoring; Approved Guideline—Second Edition

Paul J. Mountain, M.Sc., M.T. (ASCP)
Andrzej J. Knafel, Ph.D.
Suzanne H. Butch, M.A., M.T. (ASCP), SBB
Louis Dunka, Jr., Ph.D.
Rodney S. Markin, M.D., Ph.D.
David O'Bryan, Ph.D.

Abstract

NCCLS document GP19-A2—*Laboratory Instruments and Data Management Systems: Design of Software User Interfaces and End-User Software Systems Validation, Operation, and Monitoring; Approved Guideline—Second Edition* provides user interface design recommendations that make new software programs easier for laboratory personnel to learn and use. Additionally, the guideline addresses the preparation and execution of validation plans for software purchased from manufacturers; custom software commissioned by the laboratory; or in-house applications developed to collect, interpret, or report laboratory, patient, or quality control information.

NCCLS. *Laboratory Instruments and Data Management Systems: Design of Software User Interfaces and End-User Software Systems Validation, Operation, and Monitoring; Approved Guideline—Second Edition*. NCCLS document GP19-A2 (ISBN 1-56238-484-8). NCCLS, 940 West Valley Road, Suite 1400, Wayne, Pennsylvania 19087-1898 USA, 2003.

NCCLS

Suggested Citation

(NCCLS. *Laboratory Instruments and Data Management Systems: Design of Software User Interfaces and End-User Software Systems Validation, Operation, and Monitoring; Approved Guideline—Second Edition.* NCCLS document GP19-A2 [ISBN 1-56238-484-8]. NCCLS, 940 West Valley Road, Suite 1400, Wayne, Pennsylvania 19087-1898 USA, 2003.)

Proposed Guideline
November 1994

Approved Guideline
December 1995

Approved Guideline—Second Edition
February 2003

ISBN 1-56238-484-8
ISSN 0273-3099

Committee Membership

Area Committee on Automation and Informatics

Contents

Contents (Continued)

Contents (Continued)

The Quality System Approach

NCCLS subscribes to a quality system approach in the development of standards and guidelines, which facilitates project management; defines a document structure via a template; and provides a process to identify needed documents through a gap analysis. The approach is based on the model presented in the most current edition of NCCLS document HS1—*A Quality System Model for Health Care.* The quality system approach applies a core set of "quality system essentials (QSEs)," basic to any organization, to all operations in any healthcare service's path of workflow. The QSEs provide the framework for delivery of any type of product or service, serving as a manager's guide. The quality system essentials (QSEs) are:

Documents & Records	Equipment	Information Management	Process Improvement
Organization	Purchasing & Inventory	Occurrence Management	Service & Satisfaction
Personnel	Process Control	Assessment	Facilities & Safety

GP19-A2 addresses the quality system essentials (QSEs) indicated by an "X." For a description of the other NCCLS documents listed in the grid, please refer to the Related NCCLS Publications section at the end of the document.

Documents & Records	Organization	Personnel	Equipment	Purchasing & Inventory	Process Control	Information Management	Occurrence Management	Assessment	Process Improvement	Service & Satisfaction	Facilities & Safety
X GP2-A4	X	X GP21-A	X	X	X	X	X		X	X	X GP18-A

Adapted from NCCLS document HS1— *A Quality System Model for Health Care*.

Foreword

Many *in vitro* diagnostic instruments and specimen processing devices in the hospital laboratory are now computer controlled and actuated. Instrument manufacturers design and develop the embedded software that provides the functionality for these systems. The computer software presents an "interface" to the user that can make operation of the instrument a reasonable task to learn and perform.

In addition to other computerized information management systems, laboratorians in the modern clinical laboratory use a host of different computer controlled instruments, each with its own (sometimes unique) user interface. This multitude of different user interfaces affects learning, training, productivity, and potentially patient outcomes in the laboratory. Part I (Sections 3 through 10) provides guidelines for the design of software user interfaces that offer developers a common, consistent design direction for laboratory device applications.

Whether they purchase software from third-party vendors or develop the software themselves, laboratory personnel are responsible for the validation of this software. Part II (Sections 11 through 14) contains recommendations for preparation and execution of validation plans for software packages.

Key Words

Data management systems, laboratory instrumentation, process control, software design, software validation, user interface

Laboratory Instruments and Data Management Systems: Design of Software User Interfaces and End-User Software Systems Validation, Operation, and Monitoring; Approved Guideline—Second Edition

1 Scope

The scope of this document is limited to issues that affect ease of learning and the ease of use of software user interfaces. Although there is a need to improve the hardware interface between operators and instruments (e.g., keyboard, mouse, touch screen, printer, reports, voice, and light pens used when adding or removing patient samples, reagents, and waste), these topics are not within the scope of this guideline. GP19-A2 is not intended as a tool to be used in the selection, recommendation, or judgment of the suitability of specific input/output technologies, since these may change rapidly. Since it is described elsewhere,[1,2] the transfer of electronic information between information and/or automation systems (such as between a laboratory information system or laboratory automation system software and an instrument) is not the subject of this document.

This document identifies the most important factors that designers and laboratory managers should consider during the development of a new software-actuated system and when selecting a software user interface intended to improve the ease of learning and use within the clinical laboratory. Without attempting to provide a comprehensive or exhaustive discussion of software user interfaces or trying to define an identical appearance for user interfaces by describing a single, detailed design solution, this document addresses some common design elements. This discussion is intended to encourage manufacturers of laboratory instruments and specimen handling devices to develop more uniform software user interfaces within their product lines.

The primary focus of this document is the software user interface within the centralized laboratory environment. The guidelines presented in this document are not directly constructed for point-of-care, physician-office, or over-the-counter devices, although many of the principles discussed apply to these devices as well. The primary focus of this document is software user interfaces on instruments, although the guidelines also apply to interfaces on laboratory systems and other associated information systems used in the laboratory.

These design guidelines and examples are not, however, universally applicable to all laboratory systems. Implementation of a specific design depends on the size, complexity, and cost of a device, as justified by its intended use.

This document provides some simple rules to help laboratory personnel prepare validation protocols that fulfill the laboratory's obligation to test and verify the functionality and dependability of its software. This document does not advocate relieving software developers of their duty to validate the software products that they develop; GP19 offers assistance to the purchaser when no other means of validation is available. Developers should refer to sources such as IEEE sources for specific guidelines for software system validation.[3,4]

2 Introduction

This document provides guidelines for the design of software user interfaces (also referred to as "human-computer interaction") that focus on two related areas. First, it identifies elements of a user interface that are most likely to facilitate ease of learning and use. Second, it provides some direction as to commonalities in "look and feel" that further the development of an easier-to-use software interface, as well as improve its design. The subcommittee believes that GP19-A2 will initiate a dialogue between users (e.g., laboratorians) and developers of software. This dialogue is intended to bring laboratorians and manufacturers closer to an agreement as to what elements should be consistent among software user interfaces if the goals of ease of learning and use are to be accomplished.

In the modern clinical laboratory, it is necessary for clinical laboratory personnel to learn and use various user interfaces for many computer-actuated devices, including clinical instruments and laboratory information systems. The user interfaces of computer-driven devices help to manage the workflow of laboratory personnel; for this reason, they have a major effect on productivity and the effectiveness of laboratory processes. Because these interfaces have such an effect on productivity, it is important to ensure that they are easy to learn and use. This is especially true at a time when laboratorians are frequently asked to perform more tasks, as well as new tasks (e.g., CLIA compliance), with fewer or less highly skilled staff members.

In concept, "easy-to-learn" and "easy-to-use" software user interfaces have the potential to improve laboratory productivity. The reality is that user software interfaces are often difficult to learn and use. First, software user interfaces are different from one manufacturer to another; frequently, they are also different from one product to another from the same manufacturer. This makes it difficult for technicians to learn one interface and apply that learning to another interface. Second, many interfaces are difficult to use, because current knowledge in software user interface design is not applied during the development of the products. Furthermore, many products are not tested by their designers for ease of learning and use.

This document will help designers of computer-actuated diagnostic instrument systems identify those areas of a software user interface where commonalties in design, rather than product differentiation, can provide the greatest benefit to the laboratorians. GP19-A2 also provides guidelines for purchasers of computer-actuated diagnostic instrument systems that establish common benchmarks by which to judge the suitability of a potential acquisition; these guidelines move laboratorians closer to an important goal—a decrease in the total cost of laboratory operations.

GP19-A2 also offers help to laboratorians in the preparation and execution of end-user validation plans for software developed to collect, interpret, or report laboratory, patient, or quality control information from various sources, including:

- software purchased from vendors;
- custom software commissioned by the laboratory; and
- applications constructed in-house.

There was some discussion about splitting this document into two independent guidelines, because the audience for each part was different: product software designers for Part 1 and end users for Part 2. While there is some logic to that approach, it was felt that the two sections should be kept in one document to promote active discussion and interaction between both target audiences and to maximize the acceptance and utility of any software interface to laboratorians.

Software validation is a requirement of laboratories before it can be implemented, and user interface designers must understand the laboratory's need for effective, easy-to-use and easy-to-validate software interfaces. It is equally important that users understand the elements that the designers must consider in creating a user interface. In addition, because of the importance of updating this guideline in a timely way, splitting the document would have resulted in a delay in issuing both documents with little benefit to either constituency. Therefore, GP19-A2 remains as an intact guideline.

Part I: Design of User Interfaces

3 Overview

Part I addresses the key factors that affect the ease of use of a software user interface, provides guidelines for designers, and provides guidelines for evaluating the functionality of the user interfaces.

3.1 Scope

Sections 3 through 10 address the design and evaluation of software user interfaces for clinical laboratory systems, data management systems, laboratory automation systems, and specimen handling devices.

3.2 Intended Audience

Developers of software user interfaces for laboratory instruments, data management systems, laboratory automation systems, and specimen handling devices should follow these guidelines. These guidelines can be used by clinical laboratory personnel to evaluate the ease of use of software user interfaces.

To facilitate an understanding of design concepts, developers of user interfaces should read the definition, discussion, and design guideline sections, and then consider the guidelines during the development of their products.

To facilitate an understanding of design concepts and to learn how to evaluate user interfaces, clinical laboratory personnel should read the definition, discussion, and design guideline sections.

3.3 Design and Evaluation Process

A process to ensure that the design results in an effective and easy-to-operate user interface is the usability assessment.[5]

Designers should perform usability assessments throughout a product development cycle. Clinical laboratory personnel may perform usability assessments during their evaluation of the product prior to purchase.

The suggested process for performing a usability assessment is as follows:

(1) Define the target audience taking into account the demographic composure. Select the types of users who should access the user interface (e.g., novices, experts, daily users, and infrequent users).

(2) Determine which tasks should be evaluated. Include frequent and infrequent tasks, as well as critical and noncritical tasks.

(3) Develop a standard method, such as a survey, to obtain consistent feedback and comments from prospective users. Consider video taping the usability evaluation sessions.

(4) To determine pass-fail criteria, enumerate the goals and performance objectives for each task. One way to do this is by comparing the new design to an existing design.

(5) Perform the usability evaluations with the selected participants.

(6) Evaluate the results and address all usability issues that are identified.

The actual usability evaluation includes the following steps:

(1) Introduce yourself and describe the purpose of the evaluation.

(2) Inform participants that the assessment may be stopped at any time if they feel uncomfortable.

(3) Ask participants to "think out loud," i.e., verbalize their thoughts as they perform tasks.

(4) Provide the participants with any important information about the equipment.

(5) Ask the participants if they have any questions, and inform them that help will not be available during the evaluation.

(6) Instruct the participants to perform each task. During the performance of each task, observe the participants. Document errors and constructive comments made by the participants. Determine how long it takes to perform each task (if speed is important).

(7) At the end of the evaluation, ask participants to clarify their comments, and obtain their overall impressions.

(8) Compile and evaluate the results from all of the participants.

(9) Use the results.

4 Task Automation

4.1 Definition: What is Task Automation?

For each task a user performs, task automation defines what part of the task the system performs and what part the user performs. Effective task automation recognizes the strengths and weaknesses of both the user and the system and balances the design to take advantage of each.

4.2 Discussion

For a system to effectively automate tasks, it is necessary for the designer to understand the capabilities of the user and the system, the nature of the tasks, and guidelines for task allocation.

4.2.1 Human Cognitive Capabilities

Task automation involves recognizing the capabilities of users and the system, and designing a system to address these tasks in a way that takes advantage of both. An understanding of human cognitive capabilities and the human information processing system is essential to this design task.

The human cognitive system involves both short- and long-term memory. Short-term memory is the memory resource that is used for conscious thought. Short-term memory is working memory, which is used to interpret incoming information and for solving problems. Short-term memory holds the information that a user needs to remember to move from one screen to the next when performing an individual task.

Long-term memory is the memory resource that is permanent. Long-term memory is where all information and skills that have been learned are stored. Long-term memory is the memory that the user needs to remember the basic rules for operating a system.

When the user performs a task often enough to perform the same actions repeatedly, information moves from short- to long-term memory.

The human cognitive system also includes sensory organs, analogous to input and output devices. A basic controlling process, a pattern-recognition system, and knowledge and skills round out the human information-processing system.

Cognitive resources, such as short- and long-term memory and information processes, are limited. Baecker and Buxton[6] describe how the human information-processing system behaves when tasks demand more resources than are available. If additional cognitive resources would improve task performance, tasks are considered "resource limited." If performance cannot be improved with additional cognitive resources, but only with improved quality or quantity of information, the task is considered "data limited."

4.2.2 Design Guidelines: Consider the Strengths and Limitations of Short- and Long-Term Memory

When designing a system, consider both the strengths and the limitations of short- and long-term memory. Following are some design guidelines to assist with this task:

4.2.2.1 Minimize Short-Term Memory Requirements Associated With Tasks

A system should minimize the short-term memory requirements associated with tasks. If short-term memory is required, a system should decrease the total number of tasks that occur simultaneously. One way to accomplish this is by carrying over accumulated user input from screen to screen within an individual task. For example, if a sample identification number is entered during the task of ordering a set of assays on a patient sample, display the sample identification number on each subsequent screen so that the user does not have to remember which sample is being worked on. It should not be necessary to remember the specific context of information, for example: the specimen and patient identifiers should be associated so one identifier can be accessed through the other.

4.2.2.2 Design Screens and Tasks to Provide Reminders and Tools for the User

To circumvent the limitations of both short- and long-term memory, design screens and tasks in ways that provide reminders and tools for the user. Avoid designs that require the user to provide his or her own reminders and notes for performing tasks.

Whether they require the use of short- or long-term memory, provide memory aids for all tasks. Leave as few things as possible to the memory of the user. Allow users to recognize information, not recall information.

4.2.2.3 Minimize the Number of Screens Required to Perform a Task

While addressing quality of data and short-term memory concerns, minimize the number of screens a user is required to navigate when performing a task. While simultaneously striving for both readability and information accessibility, include on each screen as much of the information as necessary that is required for the user to perform a particular task. An example would be automatic display of previous results when a test is verified so that the delta check is active and without technologist intervention.

4.2.2.4 Design for a Broad Range of User Experience Levels

Design for the broad range of user experience levels. Address novice users by directing them through the task, preventing them from making mistakes, and providing landmarks and placeholders to remind them where they are in each task. Address expert users by providing built-in short-cuts for tasks, multiple paths for completing tasks, and the ability to develop macros for frequently performed tasks. Provide the ability to define user profiles that allow each user to access novice or expert features automatically.

4.2.2.5 Allocate Subtasks to Users and Systems According to Strengths and Weaknesses of Both

With an understanding of user and system strengths and weaknesses, tasks can be effectively assigned to both the user and the system. Table 1 shows the strengths and weaknesses of the human cognitive system compared to the strengths and weaknesses of the information system.

For example, tasks that require numerous mathematical calculations are best allocated to the computer, while tasks that require decision-making ability based on past experience are best allocated to the user.

Table 1. The Human Cognitive System Compared to a Computer

Computer Strengths	Computer Weaknesses	User Strengths	User Weaknesses
High-capacity memory	Limited capacity long-term memory	Infinite long-term memory	Limited capacity short-term memory
Permanent memory	Limited duration and complexity of long-term memory	Infinite duration and complexity of long-term memory	Limited duration short-term memory
Error-free processing	Limited capacity for learning	High capacity for learning	Error-prone processing
Reliable long-term memory access	Limited data integration	Powerful attention capacity	Unreliable long-term memory access
Fast processing speed	Simple template matching	Powerful pattern recognition	Slow processing speed

4.3 Task Analysis

Task analysis is a critical activity in user interface design. Task analysis involves identifying each task that a user might want to perform and breaking each task down into subtasks. Once the task is subdivided into subtasks, the information the user needs for each subtask can be identified, the possible variations and conditions that apply to the task can be identified, and the interface can be designed to assign subtasks to the system or simplify the task.

Allocating subtasks to the system requires knowledge of regulations and laboratory practice, as well as knowledge of the cognitive capabilities of users and systems. Ideally, all systems would allocate as many tasks as possible to the system and leave a few tasks that absolutely require user input to the user. The reality within the laboratory is that some tasks, even if they are easily automated, should be performed by users. For example, systems are able to validate results automatically, but regulations still require users to review and accept results.

To identify other opportunities for simplifying tasks, identify tasks as either "functional" or "operational." Functional tasks are those associated with the content of a problem. For example, ordering a test, requesting a printed report, and setting up preferences are all functional tasks. Operational tasks are those associated with how the user solves the problem with the system. Operational tasks are artifacts of the user interface. For example, pressing the buttons or making the selections required to order a test are both operational tasks.

4.4 Design Guidelines for Task Analysis and Task Allocation

Task analysis and task allocation issues have a significant effect on the design of a user interface. Table 2 lists some design guidelines associated with task analysis and task allocation.

Table 2. Design Guidelines Associated with Task Analysis and Task Allocation

Task Type	**User**	**System**
Transportation of information		●
Transformation of information		●
Simple, accurate retrieval of information		●
Algorithmic decision-making		●
Merging data		●
Pattern recognition	●	*
Multiattribute retrieval	●	*
Learning	●	*
Judgmental decision-making	●	
Creating new information	●	

* With increasing capabilities of the systems the pattern recognition, multiattribute retrieval, and learning tasks can be allocated to the computer system, but it is still the user's responsibility to validate these tasks.

4.4.1 Allocate Tasks to the User and the System According to Task Type

Table 2 provides recommendations for allocating tasks to the user and the system.

4.4.2 When Designing Tasks, Address Cognitive Resource Usage and Data Resource Availability

To simplify functional and operational tasks, it is necessary to understand the cognitive and data resources required to perform each task. Table 3 outlines the process to help the user perform functional and operational tasks.

Table 3. Cognitive and Data Resources and Their Relationship to Task Design

	Cognitive Resources	**Data Resources**
Functional tasks	Minimize	Maximize the quality of available data through screen design and organization of functionality.
Operational tasks	Minimize through consistency and the use of metaphors.	

4.5 Guidelines for Evaluating Task Automation in Laboratory Systems

- Does the system provide memory aids?
- Does the system minimize the number of screens required to perform a task, while simultaneously addressing readability and quantity and quality of information?
- Does the system address novice users by directing users through tasks, preventing errors, and providing landmarks and placeholders?
- Does the system address expert users by providing short-cuts (e.g., for repetitious keystrokes), multiple paths to complete tasks, and macro facilities?
- Does the system allocate tasks to the user and system appropriately?
- Does the system perform as many tasks as possible and leave only those tasks that require user input to the user?

5 User Control

Design the user interface to permit the user to be in control of the system at all times. The capabilities that facilitate user control are (1) the availability of interface modes for both novice and expert users; (2) the availability of a macro or scripting function (a short, simple program recorded or written by the user to automate repetitious tasks); and (3) the ability of the designer to make the complexity inherent in any program transparent to the user.

Tailoring the user interface to both novice and expert users is discussed in Section 7.2.3. The use of techniques suitable for novices permits a person to learn a new system quickly with a minimum number of mistakes, which helps to increase both the perception and reality of user control. However, productivity eventually levels off after the user becomes familiar with how the program functions. A designer should allow experts to have rapid interaction with the system using a minimum number of keystrokes or mouse clicks; yet they should also be protected from making irreversible mistakes (such as file deletion) without a warning.

One of the most effective features for increasing user productivity and permitting maximum user control is the presence of a macro or scripting capability. For example, some commercially available graphical user interfaces have built-in scripting functions that should be used when possible. Improvement in productivity comes from being able to "batch" repetitive tasks (including data input functions), as well as hard copy of archival output functions.

Making the complexity inherent in any program transparent to the user is primarily a function of the design philosophy used in the creation of the user interface. Do not present the user with more information or choices than necessary at any specific point in a program. At the same time, to permit the user to remain in control of the software, ensure that sufficient information is always immediately available on the screen.

6 Consistency

6.1 Definition: What is Consistency?

Consistency is the regularity or continuity of design features within products and between products that employ user interfaces. Keeping design features consistent is one essential way that a system can be made easier to use. Inconsistency is easily discovered and is one of the things users dislike about systems.

6.2 Discussion

Consistency is a goal in relation to all aspects of a user interface. Key areas where consistency is especially important are in task operations and interactions, language usage, and screen appearance.

Consistency in task operations and interactions is critical to ease of use and ease of learning. Users normally look for rules and patterns within the systems that they use. Systems that are consistent allow users to identify and internalize more rules, patterns, and a sense of organization. This promotes learning and recall. By identifying rules and patterns, users can predict how to perform new tasks (it is intuitive). If the system then behaves as users expect, users feel as if they have mastered the system. This sense of mastery promotes user satisfaction. Consistency in task operations and interactions limits the number of rules that users need to remember. Inconsistent systems require users to learn the basic rules plus every exception to those rules.

Designing an interface that performs consistently is not enough. Design the system, as a whole, so that it addresses individual tasks and work flow. For example, the system might define default buttons as those associated with the most frequently performed tasks. It is not appropriate to assign the default to a button associated with an irreversible action. Consistency is only useful if it is combined with good design practices.

Consistency when referring to a conceptual model (i.e., a metaphor) also promotes the formation of a consistent mental model. For example, if a system uses the conceptual model of a desktop, consistent application of the desktop metaphor allows the user to internalize that model and thus begin to understand how the system is organized and how it behaves.

Screen design and layout are two additional areas in which it is important to achieve consistency. If GUI (graphical user interface) compatibility is available, it is desirable to display reports on the screen in WYSIWYG ("What you see is what you get") form. Consistent placement of information from screen to screen allows users to predict where to find information without searching each new screen. Differences between screens that do not relate to changes in behavior expected of the user distract the user from the task of finding essential information. Make screen elements that are intended to elicit similar behavior from the user consistent in appearance from one screen to the next.

In addition, use color consistently to emphasize differences and similarities between screen elements.

Developers should adhere as closely as possible to the interface guidelines established by the manufacturer of the operating system that they are using. This usage includes the adoption of appropriate toolboxes and adherence to the guideline documents published by the manufacturer.

6.3 Design Guidelines That Promote Consistency

Consistency can be built into an interface by addressing issues of interaction, language, and screen layout. Interaction, language, and screen layout are interrelated concepts. Therefore, the guidelines for each issue are sometimes redundant.

6.3.1 Establish Conventions for Interactions and Apply Them Consistently

Consistency in interaction applies to various interaction styles, which include menus, forms, command languages, function keys, and direct manipulation.

For all interaction styles, design similar operations so that they are executed in the same way throughout the interface. Ensure that the dialogue style for different functions is consistent.

Menus enable users to navigate more quickly and locate information consistently between different screens. Inconsistencies in menu design interfere with performance and make users feel frustrated, because they perceive the system to be unpredictable. Establish conventions for menus (including screen layouts) and follow them on all menu-driven screens.

Consistency in form layout and cursor behavior is critical. Place titles and fields consistently, relative to each other. Arrange groups of fields consistently, with a consistent range of cursor movement from field to field.

For command languages, consistency in syntax is critical to the success of the interface.

For function key interfaces, make function key assignments consistent from screen to screen, from subsystem to subsystem, and within related products. It is important to consistently locate commonly used functions, such as “start” or “stop.”

For direct manipulation interfaces, similarities and consistencies in icon design are important.

6.3.2 Establish Guidelines for Screen Layout and Apply Them Consistently

Consistency in screen layout includes placement, appearance, and color.

Locate common recurring information, buttons/icons, labels, or other screen elements in a consistent manner from screen to screen.

Design screen elements that behave in a similar manner so that they have the same visual appearance. For example, ensure that all fields that take a keyboard entry have the same basic shape and appearance, and that they are sized appropriately relative to the expected entry.

With an awareness of the cultural and work-related meanings that color can bring to the interface, apply color to screen elements in a consistent manner.

6.3.3 Establish Guidelines for Language and Apply Them Consistently

Identify context-appropriate terminology that is understood by the user population, and apply it consistently across the entire interface.

Ensure that the system behavior associated with a particular term is consistent. For example, ensure that the "Start" button always has only one meaning and always executes the same task.

Ensure that the system messages and instructions follow consistent grammatical form and style.

6.3.4 Establish Techniques That Permit the User to Be in Control of the Software at All Times

The following points are provided as examples only:

- Make the use of help or information messages consistent within the two modes. Use "hot" keys for the expert mode and apply them consistently to all parts of the program.

- Use "cue cards" or "wizard" style dialogs or other prompting techniques to guide a novice through complex procedures or prompting techniques to guide a novice through complex procedures.

- Use a macro or scripting feature that is inherent to the selected operating system. If the interface does not have a built-in script language, use a third-party scripting program. Keep the complexity of the scripting at a level consistent with the skills of the user. Make sure that the user has the opportunity to select a macro or script at the point in the program where it is needed and that the user does not have to move one or more screens away to perform the selection.

- Provide all information that the user needs to perform the functions embedded in the screen directly on the screen. At a minimum, ensure that necessary information is no more than one keystroke or mouse click away, followed by immediate and simple return to the main screen, for example, by pressing the <Escape> key.

- Provide "Undo" function – see Section 9.7.2.2 for details.

6.4 Guidelines for Evaluating Consistency in Laboratory Systems

- Does the system use terminology, grammatical form, and style in a consistent manner with regard to labels, instructions, and messages?

- Are screen elements placed consistently from screen to screen?

- Is navigation through the system consistent?

- Are the rules of operation consistent?

- Is color used in a consistent and meaningful way?

- Is the conceptual model consistent?

- Do screen elements that behave similarly look alike?

- Does the system have "novice" and "expert" capabilities? Are these capabilities applied consistently throughout all parts of the program?

- Does the system guide users through complex procedures where the novice mode is used? Can the user remove these prompts?

- Is a macro or scripting function available?

- Do the screens contain excessive information or are choices presented to the user that are not needed for the functions on the screen? If so, the perceived complexity of the user interface is greater than necessary.

- If information is needed to complete a task and it is not available on the screen, is it no further than one keystroke or mouse click away? Can the user return to the main screen easily using only a single action?

6.5 Reserved Keys

To promote consistency and to build on standards established by common software user interfaces, common key definitions, i.e., "reserved keys," are recommended. New useful information can be found in the literature.[7]

The <F1> key (if available on the keyboard) should be reserved for contextual help in any situation.

The <Escape> key (if available on the keyboard) should be reserved for canceling the current operation harmlessly. This could involve allowing the user to retreat from a screen.

The <Enter> key should be reserved for (a) invoking the highlighted command or the current default command, or (b) inserting a carriage return and moving the cursor to the next line during text entry.

The <Tab> key should be reserved for (a) moving the cursor from one data entry field to the next field, or (b) moving the cursor from one command to the next. Ensure that the <Shift><Tab> combination key moves the cursor in the reverse direction.

The <Backspace> key should be reserved for (a) deleting the character to the left of the cursor or (b) deleting a selection if one was made.

7 Navigation

7.1 Definition

Navigation means knowing where one is and where one can go within a given application.

7.2 Discussion

7.2.1 Ensure that Operators Can Find Needed Functions Quickly and Easily

With little training or assistance from a "help" function, an operator should be able to find needed functions, quickly, easily, and without error.

Make the following common functions readily accessible to the operator: "help," "system status," "test selections," "data review," "maintenance," "reagent requirements," and "system configuration."

Display function options on the screen so that the operator need not memorize how to access the functions. To provide ready access, the ideal system displays as many options as necessary to the operator. At the same time, it also has a mechanism for progressively displaying infrequently performed functions. Progressive display of information prevents overloading the operator with too many confusing choices. Similar principles should be applied to keyboard functions.

Example 1: Most windows applications feature a main menu bar across the top of the display. The main menu (with pull-down submenus) allows the operator to quickly and easily access all needed functions. The shortcut information displayed for the menu entry allows the user to learn to skip the menu selection for the most frequently accessed functions.

7.2.2 Design the System to Give Operators the Ability to Know Where They Are Within an Application

Design the system to give operators the ability to know where they are within an application. Ensure that the operator is able to easily retrace the path chosen to arrive at any particular context within an application, or that he or she is able to readily see options for accessing other functions from within the current context.

When an operator chooses a function, it should be apparent that the function has been selected and that the operator is currently in that function. This eliminates confusion about what the operator selected or which item is the current focus of attention or activity.

Example 1: A typical windows application allows an operator to access and display another file while still retaining another window on the display. The current context is always displayed "on top of" the prior context. In this way, the operator knows where he or she came from and can easily return to the original context.

Example 2: In graphical user interfaces, to indicate that the selected process was successful, a selected function is highlighted.

Example 3: A "Back" button allows returning to the previous screen in case of a mistake or a changed decision.

7.2.3 Make Navigation Tools Effective for Both Novices and Experts

Make navigation tools effective for both novices and experts. Make menus available for novices and intermittent users who need to be reminded what functions exist and how to select the functions. Experts however, have memorized the functions and the tasks surrounding the functions. Make timesaving short cuts available to experts.

Example 1: Type-ahead commands or macros for experts are two effective means of improving productivity.

7.3 Guidelines for Assessing Navigation

By no means an exhaustive list, the following guidelines for assessing navigation are but a few of the possible approaches.

- Are functions required for operation present and displayed in a way that is obvious to the user?
- Are there provisions to automate repetitive functions for experts?

- Determine how long it takes an untrained operator to find five functions frequently performed within the clinical laboratory. Do operators consistently make the same errors in finding the functions?

- Determine the amount of time required to find the five frequently performed functions. If operators start at the lowest layer of a menu, are errors made in finding the functions?

- What observations or complaints do operators have related to ease of use or complexity?

A system under evaluation should perform as well as, and preferably better than, the system currently used by the laboratory.

8 Screens and Reports

8.1 Definition

The goal of defining guidelines for a user interface design is to provide users with consistent products among specialized applications. Consequently, the end-user's training is reduced and productivity is increased. Moreover, acceptance of the product is greater.

To reach this goal, design the product to fulfill key criteria. One key criterion is the requirement for simple uncluttered screens and reports, which is expressed in terms of quick access to the required information. Put the user in a position to rapidly access the data without having to decode the screen and read line by line. This "quick-access" function also ensures that the operator is able to easily switch from screen to screen. Users should be able to scroll through screens and reports and, if possible, edit them on the spot.

Also key to product design is the criterion that information presented to the user complies with quality elements, such as clarity, readability, and aesthetic appeal. That is, the information should be well organized, consistent, and should adhere to principles of good visual design.

Make interface text clear, readable, unambiguous, and free of jargon, i.e., accessible to any user in the domain for which the application was designed. Manage the aesthetic aspect through the use of basic graphic design principles that pertain to spatial grouping, contrast, and three-dimensional representation.[8]

8.2 Discussion

If an interface is designed so that the information presented is well organized, the user will be able to quickly access data without the risk of making mistakes.

8.2.1 Clarity and Readability

Clarity and readability can be achieved through the use of simple and realistic metaphors (e.g., the desktop metaphor mentioned in Section 6.2.) The metaphors should be designed very carefully. For potential "dangers" of metaphors refer to Cooper.[9] So that visual elements are rapidly comprehensible, arrange them so that functions are quickly understood and related actions are predictable. Display or report only required (essential) information on the screen.

8.2.2 Aesthetic Appeal

To achieve aesthetic appeal, make screen design pleasant and simple. Limit the number of elements; for example, do not clutter the screens with too many buttons or complex icons.

Use symbols in a way that is not confusing to the user. Make sure that the graphic language of the interface corresponds with the meanings attached to standard items as well as considers cultural differences and sensitivities.

The use of color deserves special attention. It is preferable to have a large panel of available colors from which to choose; however, do not rely on color alone to communicate essential information. For example, consider the case of a color-blind user who would be unable to access essential information if color alone were the only means of differentiating between essential pieces of information. Moreover, it is important to use contrasting colors when differentiating between various types of information.

Generally, the design and behavior of an interface should meet the expectations of the user. However, giving the user some control over specific capabilities, such as the look and the content of screens and reports, is also recommended. This allows the user to customize the software application according to his or her own style and individual tastes. Thus, the user can select only the information he or she needs and is therefore not overwhelmed by unnecessary information.

To facilitate user control of screens and reports, the manufacturer should provide the user with screen-configuration and report-formatting capabilities. As a result, the user's requirements are fulfilled and the interface complies with design guidelines and standards.

9 Error Handling: Recognition, Prevention, and Recovery

9.1 Definition

With regard to error handling, there are two goals to consider when designing a software user interface: prevention and recovery. The first goal, error prevention, is the intention to *minimize user error* using the design of the interface as a means. To do this, a designer should understand the different types of errors that users might make and why they make them. Different kinds of errors can be prevented, or at least reduced, through different design techniques. The second goal addresses the question of how to *provide easy recovery from errors* when they are made (because it is, most likely, impossible to eliminate all errors). In both cases, a context-sensitive help option can help the user accomplish tasks, while simultaneously reducing errors (see Section 9.7.3).

9.2 Discussion

Users make three types of errors: perceptual errors, cognitive errors, and motor errors.

9.3 Perceptual Errors

Perceptual errors are caused by *insufficient perceptual cues* (e.g., visual, auditory, and tactile), which result in a failure to detect important information or discriminate correctly between display objects or types of feedback. *Display objects that are visually similar* can be mistaken for one another; for example, characters that are visually similar, when presented on a low-resolution screen, might be mistaken for one another's character (e.g., B/8, Z/2, I/1).

Another type of perceptual error is the result of *invisible modes* or states. If the system can be in different states, and similar user actions have different effects in different states, then a lack of perceivable cues identifying states will result in "mode errors."[10]

Modern word processors also often have invisible modes, such as an "insert" mode or "caps-lock" mode. Users often forget they are in one of these modes and take actions that are inappropriate.

Another type of perceptual error is caused by a *failure to capture the user's attention*. For instance, important messages or data can be missed because they are visually indistinct from other parts of the display. Or, instructions can be misread because they are poorly formatted.

Finally, perceptual errors include errors due to a lack of perceivable feedback. For instance, keys can be pressed in error, or the user who presses a key can fail to perceive that fact because of a lack of auditory or tactile feedback from the keys.

9.4 Cognitive Errors

Cognitive errors are caused by *taxing the memory and problem-solving capabilities* of the human mind. For instance, some user interfaces *tax recall memory*. A command language with a positional syntax requires users to remember the proper order for specifying arguments. If they remember incorrectly, they will make an error.

Another type of cognitive error is the result of a *lack of or poor mnemonic aids*. For instance, arbitrary function key assignments, such as CMD/Shift/PF1 for moving a cursor to the bottom of the file, are difficult to remember and result in errors.

Another type of cognitive error is caused by *pressure for speed*. Users make more errors if they are pressured to perform quickly.

9.5 Motor Errors

Motor errors are the result of *requiring a high degree of eye-hand coordination*. For instance, many pointing devices require a certain amount of learned eye-hand coordination. If targets are too small, users will make errors trying to hit them.

Motor errors are also the result of *requiring motor skills in addition to those that require a high degree of eye-hand coordination*. For example, not all users have a high degree of typing skill. One estimate is that 30% of all errors made on interactive systems are simple typographical errors. Users low in required motor skills make more errors.

9.6 Error Recognition

Generally, errors can occur as a result of:

- User interaction with the system. In this case, the user is present at the computer screen, which can be used for notification about the error condition. The aspects related to this kind of error are discussed in the next chapters.

- Operation/Processing of system. In this case the user is not necessarily watching the computer screen, i.e., other means should be used for notification, e.g., visible signals independent from the computer screen or audible signals.

The error notification should be given in an unmistakable, "attention-catching" way, e.g., showing an oscillating indicator, from the time when a problem occurs until it has been acknowledged by the operator. After acknowledgment, the indicator should remain solid until the problem is corrected.

The audible signals should be easily distinguishable, e.g., multitone. The frequency of the repetition should reflect the severity of the alarm, e.g., twice per second for low and four times per second for high severity. The user should be able to adjust the volume and turn off the sound as needed. When the sound is turned off and it could be considered as the primary means of alerting the users (e.g., alarm), a visible notice should be shown on the screen to remind the user that the sound is off and they should *NOT* depend on that feature. For details, refer to Shneiderman.[11]

9.7 Guidelines for Error Prevention and Recovery

Guidelines for error handling can be divided into two categories: guidelines for error prevention and guidelines for error recovery. A designer's first goal is to minimize the number of errors that users will make. A second is to make recovering from errors as easy and fast as possible.

9.7.1 Guidelines for Error Prevention

Guidelines for preventing errors are specific to the different types of errors described previously, i.e., perceptual errors, cognitive errors, and motor errors. Understanding how these errors occur provides some insight into how to design in order to prevent them.

9.7.1.1 Minimize Perceptual Errors

One way to minimize perceptual errors or to eliminate them is to *eliminate modes*. If eliminating modes altogether is not possible, then mode errors can at least be minimized by *providing salient, visible cues for modes*.

9.7.1.2 Minimize Cognitive Errors

To minimize cognitive errors, several methods exist. *Maximizing recognition* versus *recall tasks* reduces errors because recognition memory is less error prone than recall memory.[10] Thus, for example, menus are less prone to cognitive errors than command languages. *Providing mnemonic aids* improves both recall and recognition memory. Building *consistency, rules, and patterns* into the interface reduces the learning burden and thus reduces errors. *Providing context* and *status information* to orient users prevents errors caused by misunderstanding the current context or status. *Minimizing the mental calculations and transformations* required of users reduces the errors associated with these processes in humans.

9.7.1.3 Minimize Motor Errors

Minimizing the need for high skill levels minimizes motor errors. Errors brought about by awkward motor movements are reduced by *careful key placement*, designing for the use of alternate hands for common key sequences, and minimizing the use of <Shift>, command, and other qualifier keys. Home row indicators help reduce errors caused by misaligning the fingers before typing.

Large targets and clear visual feedback reduce errors due to poor eye-hand coordination. For example, tabbing from icon to icon from the keyboard is less error prone than selecting icons with a mouse, especially if the icons are small. Tabbing requires no eye-hand coordination, while pointing with a mouse does.[8]

Minimizing the need for typing (e.g., menu selection rather than "fill-in," automatic command completion) reduces errors associated with the user's level of typing skill.

9.7.1.4 Test and Monitor for Errors and "Engineer" Them Out

If one tests and monitors for errors during the initial design process, it is possible to "engineer" them out. Careful consideration by a designer of the guidelines given here will facilitate the minimization of user errors associated with any system he or she designs.

9.7.2 Guidelines for Error Recovery

A good user interface provides fast and easy error recovery.

9.7.2.1 Provide the Appropriate Type of System Response to a User Error

Lewis and Norman[12] define six different types of possible system responses to a user error.

(1) *Gag*: Prevent the user from continuing; for example, lock the keyboard until the user takes some specific, required action.

(2) *Warn*: Note the occurrence of a potentially dangerous situation, but leave it to the user to decide how to respond.

(3) *Do nothing*: Simply do not respond to an erroneous user action; for example, do not change the attributes of an object to which the user has assigned an invalid attribute.

(4) *Self-correct*: Infer the intentions of the user and take appropriate action even if the action is not correctly specified; for instance, assume that the user means 2002 if the user types 2oo2 in the year field.

(5) *"Let's talk about it"*: Initiate a dialogue with the user to clear up the problem.

(6) *"Teach me"*: Query users to determine their intentions, and thereafter accept their previous action as legitimate; for example, accept "Exit" as a synonym for "Quit."

Lewis and Norman[12] point out that these various responses are appropriate in different circumstances. For instance, the "do-nothing" response is appropriate within a graphic, direct manipulation interface where the current state is always visually clear to the user and error messages simply add unnecessary clutter and interaction to the interface. However, the "do-nothing" response is not helpful in a command-language environment where there are fewer cues available to ascertain the current state of the system.

The "self-correct" approach can be extremely useful in handling simple typographical errors. However, in situations where users' intentions are more ambiguous, it can cause problems. Using the "self-correct" approach, the system might make erroneous assumptions about what the user really intended; this simply creates additional errors that could be more difficult to diagnose and from which it could be more difficult to recover.

The "gag" approach is appropriate for preventing serious user mistakes; however, it could be viewed as an unfriendly, "brute-strength" approach when consequences are less serious and a simple, friendly warning would do. It also tends to discourage learning through experimentation.

9.7.2.2 Provide an "Undo" Function

Often, even the simplest "undo" function is quite valuable to the user. Most systems that provide an "undo" function do so through a single keystroke or menu option. Most provide only for "undoing" the last operation, although some provide more levels of "undo." Even the ability to "undo" only the last operation can be valuable, if, for instance, that operation was deleting a 300-word passage in a document or throwing away a large file.

9.7.2.3 Provide a "Cancel" Function for Operations in Progress

Another method that is useful in error recovery is to provide a "cancel" function for operations in progress. Some operations take a long time to complete. An example is printing a large document. A user might initiate these operations in error or simply change his or her mind after starting one. Therefore, a "cancel" function is often a useful option.

9.7.2.4 Require Confirmation for Commands With Irreversible Consequences

Even if an "undo" function is available, require the user to confirm a command if it has irreversible consequences. For instance, if a user asks to erase a diskette, reformat a hard disk, or rename a file to a name already taken by another file, ask the user to confirm this request. This can be accomplished using a message such as "Do you really wish to reformat the hard disk? ALL FILES WILL BE ERASED!" It is also important to make confirmation messages explicit. A message such as "Are you sure?" is too vague to effectively warn the user. Then, to complete the action, require the user to perform an additional operation, such as a keystroke or menu pick.

9.7.2.5 Conduct Error Checking in Context but Without Interrupting Work Flow

When, for example, the fields on a fill-in form are interdependent, it makes sense to check for input errors on a field-by-field basis, as the user leaves each field. Using this example, the validity of the dollar amounts a user enters in one field depend on the dates he or she entered in an earlier field, which in turn depend on the dates entered in an even earlier field. If the initial status information is entered incorrectly, the validity of the dates and dollar amounts is ultimately affected. However, if the user does not discover that the initial status information was entered incorrectly until all three fields are filled in, then the user will have to re-enter information in all of the fields. If, however, error checking is initiated immediately after the user enters the initial status information, the likelihood that the user will make errors in the subsequent dates and dollars fields is reduced.

Often however, users who possess expert data entry and typing skills do not want to be interrupted by error checking as they work. This slows them down, and they reason that it is probably more efficient for them to work quickly, make errors, and then request error checking for all the information at once. Thus, frequency of error checking depends partly on the task and partly on user preference.

Generally, error checking can be categorized as follows:

- "On entry": convert, suppress, beep.
- "On field leave": e.g., date format entry.
- "On save": check the whole context.

9.7.2.6 Return the Cursor to the Field Where an Error Is Located and Highlight the Position of the Error

Do not require the user to interpret error messages, such as "error in date field," and then search for and return the cursor to the field where the error is located. Design the software program to return the cursor to the field where an error is located, highlight (e.g., with bold, reverse video, or color) the position of the error, and then move to additional positions as additional errors, if any, are located, highlighted, and corrected.

9.7.2.7 Allow Editing of Error Fields

Allow the user to edit error fields. Do not require a user to re-enter fill-in fields, fill-in forms, or command strings in their entirety when only one part of one field or one part of a command string is in error. Allow the user to delete and insert on a character-by-character basis and then resubmit forms and command strings.

9.7.2.8 Provide an Intelligent Means for Error Checking and Recovery

An intelligent interface might, like an intelligent human being, make an intelligent guess at what a user meant when an error was made. For instance, a command language might guess at the intended meaning of a misspelled command and give a message such as "DELATE: Do you mean DELETE? Y or N," rather than a message such as "Invalid command." Not only is the first error message more informative than the last, but also it provides a faster means of recovery from the error.

9.7.3 Context-Sensitive Help

Make more detailed error assistance available through *context-sensitive help*. Even when error reporting is well designed, some users might need more detailed information than can, or should, fit onto the display where an error occurs. A single keystroke or menu option can provide this detailed information within the error field in which the cursor is currently contained. Help can be presented in "Tool tips," "What's this?", pop-up boxes, windows, separate screens or as "on-line" (Internet) help. The proper mechanism should be chosen for the related context. Also, make it easy to return from the "help" display to the field that contains the error.

Context-sensitive help can help users learn how to use a computer software system more effectively. An effective feature for system designers to consider using is one that includes separate "levels" of help geared toward novice or expert users.

To ensure its value to the user, make context-sensitive help relevant to the task at hand. Provide general information that gives the user sufficient background information to make intelligent decisions about the system that he or she is using.

Provide specific information about all significant elements of the display that are not intuitively obvious to *all* users. If reference is made to a menu, describe all of the entries on the menu. If reference is made to a data entry screen, describe each field in which data is to be entered. Also, describe the effects of all of the commands that can be used from within that screen. Include descriptions of possible side effects and a discussion of implications to operation of the system as a whole if certain choices are made.

9.7.4 Design Error Messages to Facilitate Error Recovery

As suggested in the study by Shneiderman,[13] the quality of error messages can affect error recovery. Table 4A below lists several guidelines for designing error messages; Table 4B addresses these guidelines for designing error messages in greater detail.

Table 4A. Guidelines for Designing Error Messages

• Be descriptive but concise.	• Avoid using exclamation points.
• Do not mislead.	• Avoid using violent and hostile words.
• Be prescriptive.	• Use a consistent grammatical style.
• Design detail according to user knowledge and experience.	• Use a consistent grammatical style.
• Take the blame.	• Do not anthropomorphize.

Table 4B. Guidelines for Designing Error Messages

Original	Improved	Comments
Filing error	Disk full	The original message is vague; however, the improved message provides more specific information to aid error recovery. Note that there are no more words in the improved message; it is just as concise but more descriptive.
File not found	Missing filename extension	If the error is failure to specify a file extension in a file name, then the original message is misleading. Consequently, the user might assume that a file was deleted or moved to a different directory, when in fact it still exists.
Disk full	Disk full. Use "Save As" command to save to another disk.	The original message is descriptive but not prescriptive. That is, it tells the user what is wrong but not what to do about it. The improved message helps the user determine how to recover from the error.
Error in DILUTION field	Error: DILUTION FIELD range is 4 to 16. No leading zeros.	For users low in semantic and syntactic knowledge, the original message is too vague. These users need the additional semantic information ("range is 4 to 16") and syntactic prompting ("no leading zeros") provided in the improved message. Users high in semantic or syntactic knowledge do not need one or both of these types of information. Design error messages to match user's knowledge and experience.
Bad input	Unrecognizable command	The original message implies that the user has made an error. The improved message implies that the computer is at fault. The improved message is less intimidating to novices and less offensive to the user in general.
Unrecognizable command!!!	Unrecognizable command	Use of exclamation points is often unnecessarily dramatic. As in the original message, the exclamation points imply that the error is inexcusable. They have the potential to alarm novices and offend experts.
Disastrous Fatal Abandoned Illegal Invalid Catastrophic Terminated Bad Aborted Killed Failed	Halted Unrecognized Cannot accept Could not execute	The text of the original message is typical of many computer error messages. They are violent, melodramatic, and hostile. They can be intimidating to novices and offensive to nontechnical experts. Other words, such as those in the improved example, express the same meaning using a more benign, less threatening tone.
Unacceptable Run canceled	Cannot accept Cannot recognize Cannot run	In each of the three original messages, the grammatical structure is different while the grammatical structure of the three improved messages is the same. The meanings, however, are the same in the two sets of messages. Grammatically consistent messages are easier to read and understand.
(bottom of screen)	(in window) (on top of screen) (next to error field)	Place messages where the user's eye is likely to notice them. On screens of a traditional size, always placing the error message at the bottom of the screen might be appropriate because the message is always well within the boundaries of the user's peripheral vision. On larger screens, however, the bottom of the screen could be outside the boundaries of peripheral vision, depending on where the user's eye is focused. In this case, error messages are more noticeable if they appear closer to the current focus of attention, such as in the current window or next to the field in which the error occurred.
Sorry Mary, I can't accept that command.	Cannot accept command	In the original message, addressing the user by name and using a humorous or conversational tone gives human attributes to a nonhuman entity (the system). Although a designer might be amused by designing humanlike qualities into a user interface, the result often deceives, confuses, and misleads the user. It can make a system seem rather unlike the tool that it is, which can weaken the user's sense of responsibility for intelligent use of the tool.

10 Security

Generally, the purpose of an information security is to provide a set of rules, measures, and procedures that determine the physical, procedural, and logical security controls ensuring confidentiality, integrity, and availability of the information.

The responsibility of the user's organization is to define a security policy covering:

- Data sensitivity classification.
- User registration.
- Privileged access.
- Password control and management.
- Log-in procedures and time-out procedures.

User interface should comply with requirements derived from:

- EU Directive 95/46/EC (1995).
- 65 Federal Register 82461-82829 (2000) (codified at 45CFR160-164).
- Pub L No. 104-191.
- 63 Federal Register 43241-43280 (1998) (codified at 45CFR142).
- 62 Federal Register 13464 (1997) (codified at 21CFR11).

The implementation of the system and the user interface should address:

- log in of user, including user authentication and authorization.
- tracking of changes to the system configuration made by user.
- automatic protection after period of inactivity at the User Interface.

The authentication can base on special access keys, biometry (e.g., fingerprints), or passwords. In case of using passwords, an automated password management should be implemented where possible (length of password, periodical change of password, protection against reusing of passwords within 12 months, etc., for example).

Part II: Systems Validation, Operation, and Monitoring

11 Overview

Part II of this guideline addresses the design, preparation, documentation, and use-validation plans for systems used within the clinical laboratory.

11.1 Scope

Sections 11 through 14 address validation plans (design, preparation, documentation, and use).

11.2 Intended Audience

When performing validation of systems, system administrators and supervisory personnel should follow these guidelines. Vendors of clinical laboratory systems should refer to this document during the preparation of recommendations for validating new or revised software.

The vendor is obliged to validate the system according to requirements of the regulatory bodies and software/system engineering practices. This practically means that the vendor is responsible for ensuring correct functioning in all specified use-scenarios.

The user of the system is responsible for making sure the system is functioning properly in his or her environment and use-scenario (assuming that this scenario belongs to the set intended for the system by the vendor). The user should validate this system according to guidelines supplied by the vendor. Many systems support user-defined functionality (e.g., macro procedures, custom calculation formulas). If the user customizes the system, then the validation of the proper function of this user-defined function is the responsibility of the user.

11.3 Definitions[a]

Access point – A function or step in the operation of software that allows the "inputting" of information; **NOTE:** An access point can influence the behavior of the system.

Acceptance criteria – A defined set of conditions that must be met to establish the performance of a system; **NOTE:** These conditions define the acceptability of the software from the user's perspective.

Critical point – A step in the operation of software that is essential to the quality of the function or task; **NOTE:** A critical point can influence the behavior of the system's user or be a system-performed calculation, interpretation, or algorithm.

Significant change – A change that affects the quality or accuracy of the system.

Test case – Description of a program, function, requirement, or condition being tested, the data to be entered, and the outcome expected.

User – The facility or person using the system.

Validation – Confirmation through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled (ISO 9000:2000 [3.8.5]).

Vendor – The hardware/software firm or personnel responsible for development, installation, training, or maintenance of the product or service.

11.4 Resource Requirements

Determining the resource requirements for a validation effort is a difficult but necessary task. Because all systems and laboratories differ, each validation plan is unique.

To assess resource requirements in the laboratory, record and track the resources used in a small part of a larger validation plan, or in a small project. Keep track of the time and personnel needed to write and run the test cases. Use this information for future validation planning.

12 Planning

To ensure thorough inspection of software and that acceptable criteria are met, a plan that details the process of validating and accepting software used in the clinical laboratory is required. To establish a plan for the validation process, the following procedures and documents are recommended.

[a] Some of these definitions are found in NCCLS document NRSCL8—*Terminology and Definitions for Use in NCCLS Documents.* For complete definitions and detailed source information, please refer to the most current edition of that document.

12.1 Project Management

Identify all internal and external resources that are required to complete the user validation project, including preparation, implementation, and follow-up. Specify all personnel and material resources, as well as timelines for project tasks in the plan. Clearly define the responsibilities and authority of personnel involved in the validation project. If appropriate, include a project budget.

12.1.1 Personnel

Depending on the size and nature of the different functions and responsibilities that exist within any validation project, personnel resources will vary. In practice, one or more of these responsibilities may reside with a single person.

12.1.1.1 Project Manager

The project manager is the person responsible for specifying and managing project resources. This includes ensuring that budgets and timelines are met and reporting progress to senior management.

12.1.1.2 Vendor Liaison

The vendor liaison is the person who represents the vendor or supplier of the system undergoing validation. This person's responsibilities include the resolution of problems and discrepancies identified during the validation process. In some situations, the validation requires participation of more vendors (e.g., interface of Lab Automation System to Lab Information System). The good practice is to involve all participating vendors in the entire process in order to avoid potential misunderstandings and "finger-pointing."

12.1.1.3 User Representative

The user representative is the person responsible for "signing off" on the final results of the user validation process and, ultimately, for determining the acceptance or disapproval of a system.

12.1.1.4 User(s)

The user(s) is the person (or persons) responsible for specifying the acceptance criteria and for implementing the User Validation Plan.

12.1.2 Materials/Resources

Specify the materials/resources that are required for implementation of the test plan. These may include but are not limited to:

- Test systems and equipment.
- Facility resources (space, utilities, direct and indirect [e.g., via Internet] interfaces among involved systems).
- Usage costs.
- Consumable supplies (paper, printer ribbon, storage media, and clinical supplies).

12.1.3 Timeline

Prepare a timeline by describing the individual tasks associated with the user-validation project, their projected resource requirements, and required or projected completion times. The level of detail used depends on the scope and complexity of the project. To ensure that time and budgetary goals are met, monitor and update the timeline during the project.

12.2 Assess How the Project Will Affect the Operation of the Institution

During the planning phase, assess how the user validation project will affect the operation of the institution. Potential items to consider include:

- The effect of diverting personnel and material resources to the validation project.
- If a "live" environment (i.e., the system in operation) is used during validation, the steps that are necessary to ensure safe, continued, routine operations.
- The effect on the institution if project goals (budget, timeline) are not met.
- The effect on the institution if the system is ultimately not accepted.

12.3 Specifications and Documentation

Specifications and documentation that describe the requirements for software, its major features, and an overall configuration scheme provide a baseline set of information about the intended performance of the software. Use additional information, as supplied by the vendor, to expand the performance definition whenever possible. See Appendix A.

12.3.1 "Request For" Proposal

Typically, an institution develops a "Request for" proposal, which is a set of written requirements for any major capital purchase. "Request for" proposal is used to define specific software requirements for the clinical laboratory and it is delivered to the vendor. To establish the opening criteria for the performance of a system, those responsible for purchasing software should adhere to this policy. The "Request for" proposal may include but is not limited to the following components:

- Major features and functions.
- Cost.
- Service and upgrade terms and policies.
- Hardware requirements.
- Performance characteristics, including database access and screen refresh times, and database size (e.g., number of samples).
- Compatibility with other institutional software systems (i.e., hospital information system [HIS], laboratory information system [LIS]).
- Data interchange formats supported (e.g., third party application, ASTM, and Health Level 7 [HL7] standards).

12.3.2 Vendor-Supplied Documentation

Product brochures, advertising, and all types of material provided by vendors of computer software are permissible as documentation for the validation plan. These documents, along with any other performance and feature information, provide the basis for vendor-supplied documentation.

12.3.3 Vendor Specifications

Manufacturers of computer software should document their products in such a way that purchasers can easily adopt a strategy to incorporate their products into the laboratory. Ideally, facility requirements include the same components called for in the request for proposal. Additional documentation may include:

- Operational and environmental specifications.
- Labeling produced as operators' and/or users' manuals. Most operators' manuals include a section on "system specifications," which is helpful in developing a validation plan.
- Diagrams, process flow charts from installation procedures, and other descriptions that provide information about the interaction of modules, features, and functions.
- Training documentation in the form of "help" functions, tutorials, and manuals.

12.3.4 Operator's Manual

The operator's manual contains most specification information about a product, including process flow charts and feature descriptions. It is possible to derive system access points and a general testing strategy from the specification information in the operator's manual.

12.3.5 Configuration

Assess the installed system configuration to determine an overall testing and validation plan:

(1) Determine specific features and options that are enabled/disabled.

(2) To specify an overall hardware configuration diagram, lay out the number of processors, printers, terminals, external interfaces (HIS, LIS) and other peripherals.

12.3.6 Communication Interface Specifications

Communication interface specifications identify the type of information that can be exchanged between systems. The specification should not only include the description of the communication protocol (i.e., format of the data), but also the semantics (i.e., the meaning of data elements, e.g., coding of flags with their interpretation). If the plan is to integrate systems within the laboratory environment, this information is critical in planning for the validation of this access point.

13 Testing

Testing to certify the performance of a system is the key component of the validation plan. Software testing is designed to test the features and applications of a system both functionally and structurally.

13.1 Strategy

The overall testing strategy is defined by the level of documentation supporting the software and the needs of the laboratory. Often, software installation takes more time than anticipated and the need to make the system operational becomes urgent. Regardless of the pressure on the laboratory to get the system "up and running," it is important that a testing strategy involve the amount of testing necessary to determine the overall performance of the system. Measure this performance against the requirements set forth before purchasing the system, as well as the individual needs of the laboratory.

Determining the extent of testing of a module or function is dependent on the risk associated with failure. The greater the harm associated with the failure, the more extensive the testing. Functions such as an interpretation of a result may be tested using all options available while a calculation will be suitably tested using carefully selected results to include boundaries and special cases. Quality assessment tools such as Failure Mode Effects Analysis (FMEA) may be useful in assessing the points of failure and the impact of the failures. The amount of testing should be related to the potential for a serious failure.[14]

13.2 Approach (Component vs. System)

Validation of a system focuses on testing from a workflow perspective in conjunction with the routine processes within the laboratory. Testing may include normal testing, boundary testing, invalid or special case testing, as well as parallel testing (running the proposed system in parallel with the existing system for a specified period of time). Parallel testing may assist in verifying that the new system produces the same (or better) results than the existing system, but cannot be the sole form of testing as it focuses only on cases presented during normal test runs.

For further discussion of various approaches to testing, see Steane et al.[15]

13.3 Environment (Active vs. Static Tests)

In most cases, it is not possible to perform validation on-line with an operational system. To do so would jeopardize the integrity of other processes that might be affected by an abnormal system. In this case, an off-line, static test is required with simulated conditions from a test database.

An off-line test affords the opportunity for greater experimentation and better control of the overall testing process. It is recommended that off-line, static testing be performed as much as possible to validate the operation of the system.

While static testing allows better control of the testing process, it does not always stress the system to its breaking point; therefore, a certain amount of dynamic testing is required. Dynamic tests that stress all functional areas are well worth the time and effort to assure a quality system. Users should understand that stress testing is more than using large amounts of data, processes, and concurrent users. Stress testing involves an in-depth analysis of the possible breaking points of the system beforehand and development of test cases that adequately challenge these areas. For a further discussion of stress testing, refer to Beizer.[16]

The dynamic testing can be supported by real-time simulators, allowing a setup of the environment and conditions comparable to the routine case.

13.4 "Sign-Off" and Approvals

Ensure that all necessary documents and records are signed and dated by the appropriate personnel. For example, make sure that each test case is signed and dated by the person who executed the test.

13.5 System Access Points

Because data input and output greatly affect the performance of a system, a recommended testing plan identifies and interrogates all system access points. Examples of system access points include the following:

- The order entry process in a clinical laboratory information system.
- Bar-code reader input from a hand-held device used in conjunction with a blood bank program.
- Remote patient report functions invoked from a hospital nursing station.
- Uploading of patient medication information from the pharmacy application of an LIS.
- Data entry into a third party application as part of the HIS.
- Data transfer from one system to another (e.g., patient orders and test results).

13.6 Data Collection Methods

Collect data according to the defined test plan and specific test cases (see Section 13.8). Retain all printed material (reports, tapes, test case forms, and logs). Retain each test case with all attachments as validation of the test or indication of the action taken if a "fail condition" is received.

Then archive each test case in accordance with the overall validation plan and with the individual requirements of the laboratory.

13.7 Acceptance Criteria

As a proof and measure of performance, establish acceptance criteria. Without these criteria, validation of a software system is a subjective analysis of the information obtained from the validation plan.

Minimally, it is recommended that acceptance criteria challenge the following points:

- User requirements and the "Request for" proposal were met.
- The overall validation plan (and all test cases) were passed. Failed cases were explained or otherwise accepted as noncritical, and potential risks were identified.

13.8 Test Plan

13.8.1 Test Plan Resources

In preparing a test plan, other resources can be used to provide useful information, but they do not replace the written plan or test cases. Other resources include:

- Vendor manuals and recommendations.
- Standards publications (i.e., a list of the standards established by other agencies).

13.8.2 Test Case Design

Test cases should thoroughly challenge the ability of the system to meet specified user requirements. Test cases and procedures should be developed based upon laboratory procedures.

(1) Management

Test case design management needs to ensure that:

(a) A uniform test case format will be used; and

(b) A system for identification of test cases will be implemented in order to ensure a high level of version and document control as well as facilitate traceability of the test case to a corresponding user requirement.

(2) Comprehensiveness

To ensure that testing challenges all aspects of the system which pertain to user requirements, an analysis should be done to identify all system inputs and outputs as well as process paths. System documentation should provide most of this information.

(a) Inputs and Outputs. Those access points where data is passed to or from the system. These include, but are not limited to, keyboards/displays, instrument interfaces, interfaces with networks or external computers, bar code/card readers, and modems.

(b) Process Paths. These include the range of possible operations that can be performed by the system on data including decision points or calculations.

Once these have been identified, failure modes and the impact of those failures can be identified. Conditions to be considered should include operator error, communication errors, and failures induced by external factors such as power failures. From this information, a list of test cases can be developed.

(3) Testing Methods

Test cases should thoroughly challenge the system's ability to meet user requirements. The following test methods should be applied:

(a) Boundary Testing. Where there is a range of valid inputs, values should be tested which are within the range of valid entries, at the limits of the range of valid entries, and beyond the range of valid entries.

(b) Stress Testing. Where a system has a requirement to perform at a certain level, test cases should be developed to challenge the system at and beyond those levels.

(c) Input Testing. Testing of data input functions should include valid values, invalid values, null and zero values, and values of an incorrect data type.

(4) Test Case Content

Each test case should contain the following elements:

(a) the system, module, and/or component being tested.

(b) the requirement being tested.

(c) a description of the scenario.

(d) dependencies-what other test cases depend on for the successful completion of this test case.

(e) date and time of the test.

(f) the person(s) executing the test case.

(g) number of repetitions.

(h) the test procedure, including an explicit description of the system configuration parameters, input data and the expected outcome.

(i) required documentation-printout, screen capture, etc.

(j) acceptance criteria.

(k) a place to designate whether the test passed or failed.

(l) next action in the case of both pass and fail.

(5) Regression Testing

Each test case should include a list of dependent test cases that must be repeated in the event that the current test case fails. These test cases would be repeated once the deficiency is resolved.

13.9 Error Analysis and Handling

Results of the validation protocol and each individual case shall be examined individually. Failed test cases should be held for review after completion of the first pass of testing, unless the error is obvious and can be rectified immediately (e.g., printer off-line or other operator induced error). Failed test cases or unexplained conditions need to be analyzed and determined to be noncritical for the performance of the system to allow a pass condition. Otherwise, failed test cases may be repeated to confirm the initial results. It is the responsibility of the user(s) to ascertain what errors comprise a total system failure and to assess the impact on the go/no-go decision. Data analysis should occur as a group with all participants in agreement. It is recommended that the vendor participate in this review if possible.

13.10 Evaluation of Results

Having completed the validation protocol, the final step in the validation plan involves an evaluation of all data and information gathered during the process and a decision about the performance and acceptability of the software system. To ensure that all conditions are met, discuss all errors, discrepancies, and unexpected results thoroughly.

13.10.1 Evaluation

Once all validation results are compiled, conduct a roundtable review of the results. Participants in the roundtable review should include all personnel involved in the validation, as well as the vendor(s). For historical purposes, retain a report summarizing the findings of the group by the laboratory. In the evaluation, address the following key areas:

- Results of the validation protocol and test cases.

- Discussion of all errors, discrepancies, and unexplained occurrences, as well as actions taken and conclusions reached.

- Implications of all errors and discrepancies.

- Comparison of findings with acceptance criteria.

- A checklist of required documentation (operator's manual, service and training documentation) received.

- Completion of all ancillary duties (e.g., archival of software).

13.10.2 Final Approval

To complete the evaluation process, it is necessary to obtain affirmation of the validation of the system and final written approval from all participants. It is important that the vendor participates in this process and approves the validation of the system. Agreement between the vendor and the user facilitates any future communication and discussion about possible errors and additional changes to the system.

13.10.3 Go/No-Go Decision

The decision to actually implement the software in the routine operations of the laboratory—the "go/no-go" decision—is independent of the approval process. In some cases, although final approval has already been granted as a result of the validation process, implementation is delayed because of further expansion in the laboratory or because of cost. Conversely, a system might be deficient in its ability to meet select user requirements, but it might also contain additional benefits that outweigh this deficiency.

If, however, the system is not installed as a routine operation available throughout the laboratory, all of the features should be eliminated from the system. Each module or program should be removed from working directories or secured from user access.

14 Operation and Monitoring

14.1 Security

Implement security procedures, consistent with the sensitivity of the data, that limit access to the system. To ensure the confidentiality of information contained within the system and to protect the system from unauthorized use, controlled access is necessary. Data to be controlled includes directories, files, archive disks, or tapes and any other media containing sensitive information. The end user should verify the security provided by the vendor functions properly in the end user's environment.

The means of the data transfer, e.g., e-mail, web-access, etc., should be covered by the security measures (e.g., encryption, digital signatures) as described in:

- EU Directive. 95/46/EC 1995.
- 65 Federal Register 82461-82829 (2000) (codified at 45CFR160-164).
- Pub L 104-191.
- 63 Federal Register 43241-43280 (1998) (codified at 45CFR142).
- 62 Federal Register 13464 (1997) (codified at 21CFR11).

14.1.1 Methods for Controlling System Access

Depending on the intended use of the system within the laboratory, provide a method for controlling access to the system. For example, simple classification/restriction of users according to different levels of access is one possible method of controlling access. Implementation of the use of passwords is another alternative.

14.1.2 Surveillance/Audit Trails

The user should verify the implementation of security by vendor in accordance to his institutional requirements. Audit trails are required for some transactions such as electronic signatures.[b]

14.2 Records

14.2.1 Standard Operating Procedures

Make standard operating procedures (SOPs) available for all of the procedures performed during the installation, operation, and maintenance of a system. The procedures required include:

- Validation procedures.
- Normal operations.
- Routine data maintenance procedures.
- Disaster plans.
- Security procedures.
- Training procedures.
- Change control.
- Archiving and restoration of purged records.
- Compliance audits.

Have SOPs approved by a person or persons with knowledge of the system and vendor recommendations.

Maintain the procedures according to the policies established for other laboratory policies and procedures. Refer to NCCLS document GP2—*Clinical Laboratory Technical Procedure Manuals*.

Use a standard format. Define the retention period and make it consistent with other laboratory record storage requirements.

14.2.2 Software Validation Records

Maintain software validation records in accordance with the overall validation plan. It is recommended that records contain the following information:

- Acceptance criteria and user requirements.

- A validation protocol.

- Individual test cases with appropriate attachments as part of the overall validation plan.

- Logs indicating notes from testing (if not part of individual test cases).

[b] In the U.S., refer to Title 21 Code of Federal Regulations (21CFR11), Section 11.10(k)(2).

- Corrective actions taken as a result of failed tests or unexplained occurrences.
- Analysis and review of results.
- Final decision (“go/no-go”) and approvals.
- Description of change-control and audit procedures to be followed.
- Certification by vendor of installation process.
- Affirmation that an archived copy of installed software is available.

14.2.3 Document Retention

Retain procedure manuals for at least five years after revision.

Maintain validation documentation for the system in use throughout the life of the instrument.

Maintain documentation of changes in the operating parameters of the system for the length of time a version of the system is in use.

Maintain documentation of changes in test data for the length of time test data are retained.

14.2.4 Maintenance

User testing should be repeated whenever an event or modification has occurred which may impact the operation of the system. These may include but are not limited to:

- Installation of a hardware or software upgrade.
- Preventive maintenance of hardware or software.
- Repair of a hardware or software defect.
- A change or repair to an external instrument or system which is interfaced to the system.

The scope of the retesting is dependent upon the system components affected and the risk involved due to the occurrence. The number and nature of the test cases to be repeated should be determined based on recommendations of the vendor and/or service organization, as well as a thorough internal assessment of the situation and its potential impact on the system's meeting user requirements.

14.3 “Change Requests”

Modifications to a system require control in a well-defined manner. Both the vendor and the user bear the responsibilities involved in the modifications; ideally, they are agreed upon at the time of installation. Typically, a “change request” is initiated when the user requires a new feature or functionality in the system, perhaps to maintain compliance with changing regulatory requirements. Periodically, changes are mandated because of an error or “bug” in the system.

A recommended procedure for change-control includes the following responsibilities, which have implications for both the user and vendor:

14.3.1 User Responsibilities

(1) Before the action, assess "need vs. want" and the potential effect on all areas of the laboratory. Confirm receipt of all of the necessary approvals.

(2) Compile written requests that encompass the following areas: description of modification(s) to the system, any screen modifications; effect on the file system and other functions; proposed output; and hardware modifications (if any).

(3) Create an installation plan that addresses the conversion of affected data and tables, timing, and the definition of the validation protocols. Keep records of all requests sent to manufacturers.

(4) Create proposed secondary training and instruction plans and address their impact (if necessary).

(5) Notification to all internal and external (peripheral) users of the intended change. An approximate schedule for instituting the change and a brief description including benefits ought to be provided.

14.3.2 Vendor Responsibilities

(1) Accept each request from the user.

(2) Process each request according to in-house SOP for "change requests."

(3) Notify the user of the implications of and decision about each "change request."

14.4 Training

Describe the training program related to the use of the software. The training program should include classification of potential users of the system and the level of training required for each. Depending on the scope and complexity of the system, and on the number of people who need training, one or more persons within the laboratory can be designated as trainers. These persons can participate in vendor-supplied training and, in turn, train the rest of the users in the laboratory. It is the responsibility of the laboratory to ensure that training is adequate and effective, and to maintain documentation that training has taken place.

The training program should include the following elements:

14.4.1 Initial Training

Initial training is performed when a system is first installed. The training program may include a vendor-supplied part, as well as a laboratory-specific part.

14.4.2 After a Change

After a change, describe the procedure for training that should be completed as a result of modifications to the original system. In most cases, this will not be as comprehensive as initial training.

14.4.3 New Users

Describe the procedure for training new staff members or users. The plan may dictate that the new user(s) receive vendor-supplied training or in-house training.

14.4.4 Proficiency

Describe the procedures by which proficiency is to be demonstrated and documented. These procedures should include the methods for evaluation and they should cover (1) initial competency evaluation, (2) competency evaluation after a change to the system, (3) routine, periodic competency re-evaluation, and (4) competency evaluation after retraining as a result of competency deficiencies.

14.4.5 Documentation

Describe the procedures by which training documentation is to be maintained. This includes the maintenance of vendor-supplied and internally developed training materials, as well as training records.

Minimally, training records should include (1) the name and classification of the user, (2) the date and type of training, (3) criteria and proof of acceptable performance, and (4) identification/signature of the trainer or supervisor.

14.5 Audits

Regularly scheduled audits are recommended as an integral component of a laboratory quality assurance program and as a follow-up to the installation of new or modified software. A quality assurance program for validation of systems includes documentation that adequately describes an audit process or program including a schedule for periodic testing and elements to be reviewed. Audit documentation is minimally defined as:

- A definition of elements to be tested.
- Testing plans to validate and ascertain that features within the software perform as planned and without error.
- Acceptance criteria that, when compared with the results of a testing/validation plan, show whether all requirements were met and identify actions to be taken if these requirements were not met.

All laboratories are urged to implement an audit policy that routinely validates the performance of their systems in the laboratory according to the guidelines listed above.

For additional information about audit practices, see Steane et al.[15]

References

[1] ASTM. *Specification for Transferring Information Between Clinical Instruments and Computer Systems.* E1394-91. Philadelphia: ASTM; 1991.

[2] ASTM. *Specifications for Low-Level Protocol to Transfer Messages Between Clinical Laboratory Instruments and Computer Systems.* E1381-95. Philadelphia: ASTM; 1995.

[3] The Institute of Electrical and Electronics Engineers, Inc. *IEEE Guide for Software Verification and Validation Plans*. IEEE 1059-1993. New York: IEEE; 1993.

[4] The Institute of Electrical and Electronics Engineers, Inc. *IEEE Guide for Software Verification and Validation Plans*. IEEE 1012-1998. New York: IEEE; 1998.

[5] Neilson J. *Usability Engineering*. New York: Academic Press; 1992.

[6] Baecker R, Buxton W. Cognition and human intuition processes. In: Baecker R, Buxton W, eds. *Readings in Human Computer Interactions.* Los Altos, CA: Morgan and Kaufmann Publishers Inc. 1987:207-218.

[7] Microsoft Corporation. Appendix B, "Keyboard Interface Summary." In: *Microsoft® Windows® User Experience*. Redmond, WA: Microsoft Press; 1999.

[8] Mayhew DJ. *Principles and Guidelines in Software User Interface Design*. Englewood Cliffs, NJ: Prentice Hall; 1992.

[9] Cooper A. *About Face: The Essentials of User Interface Design*. Foster City, CA: Programmer's Press. 1995:53.

[10] Norman DA. Design rules based on analyses of human error. *CACM*. 1983;26(4):254-258.

[11] Shneiderman B. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. 3rd ed. Boston, MA: Addison-Wesley Publishing; 2002.

[12] Lewis C, Norman DA. Designing for error. In: Norman DA, Draper SW, eds. *User Centered System Design*. Hillsdale, NJ: Lawrence Erlbaum Associates. 1986:410-432.

[13] Shneiderman B. *Pracniques: Designing Computer System Messages. CACM*. 1982:25(9):610-611.

[14] McDermott RE, Mikulak RJ, Beauregard MR. *The Basics of FMEA*. New York: Productivity Inc.; 1996.

[15] Steane S, et al. *Responsibilities in Implementing and Using a Blood Bank Computer System*. Arlington, VA: American Association of Blood Banks. 1989:3.

[16] Beizer B. Background stress testing and performance. *Software System Testing and Quality Assurance*. New York: Van Nostrand Reinhold; 1984.

Additional References

For additional information on user interfaces or software validation, the following documents are recommended.

ASTM. *Standard Guide for Selection of a Clinical Laboratory Information Management System.* E792-95e1. West Conshohocken, PA: ASTM; 1995.

ASTM. *Standard Guide for Documentation of Clinical Laboratory Computer Systems.* E1029-01. West Conshohocken, PA: ASTM; 2001.

ASTM. *Standard Specification for Transferring Clinical Observations Between Independent Computer Systems.* E1238-97. West Conshohocken, PA: ASTM; 1997.

ASTM. *Standard Guide for Description of Reservation/Registration-Admission, Discharge, Transfer (R-ADT) Systems for Electronic Health Record (EHR) Systems*. E1239-00. West Conshohocken, PA: ASTM; 2000.

ASTM. *Standard Practice for Reporting Reliability of Clinical Laboratory Computer Systems.* E1246-01. West Conshohocken, PA: ASTM; 2001.

ASTM. *Standard Guide for Construction of a Clinical Nomenclature for Support of Electronic Health Records.* E1284-97. West Conshohocken, PA: ASTM; 1997.

ASTM. *Standard Guide for Rapid Prototyping of Computerized Systems.* E1340-96. West Conshohocken, PA: ASTM; 1996.

ASTM. *Standard Guide for Content and Structure of the Electronic Health Record (EHR).* E1384-99e1. West Conshohocken, PA: ASTM; 1999.

ASTM. *Standard Specification for Use of Bar Codes on Specimen Tubes in the Clinical Laboratory.* E1466-92. West Conshohocken, PA: ASTM; 1999.

ASTM. *Standard Specification for Transferring Digital Neurophysiological Data Between Independent Computer Systems.* E1467-94. West Conshohocken, PA: ASTM; 2000.

ASTM. *Standard Specification for Coded Values Used in the Electronic Health Record.* E1633-00. West Conshohocken, PA: ASTM; 2000.

ASTM. *Standard Guide for Functional Requirements of Clinical Laboratory Information Management Systems.* E1639-01. West Conshohocken, PA: ASTM; 2001.

ASTM. *Standard Specification for Representing Clinical Laboratory Test and Analyte Names.* E1712-97. West Conshohocken, PA: ASTM; 1997.

ASTM. *Standard Specification for Transferring Digital Waveform Data Between Independent Computer Systems.* E1713-95. West Conshohocken, PA: ASTM; 1995.

ASTM. *Standard Guide for Properties of a Universal Healthcare Identifier (UHID).* E1714-00. West Conshohocken, PA: ASTM; 2000.

ASTM. *Standard Practice for an Object-Oriented Model for Registration, Admitting, Discharge, and Transfer (RADT) Functions in Computer-Based Patient Record Systems.* E1715-01. West Conshohocken, PA: ASTM; 1998.

ASTM. *Standard Guide for View of Emergency Medical Care in the Computerized-Based Patient Record.* E1744-98. West Conshohocken, PA: ASTM; 1998.

ASTM. *Standard Guide for Electronic Authentication of Health Care Information.* E1762-95. West Conshohocken, PA: ASTM; 1995.

ASTM. *Standard Guide for Properties of Electronic Health Records and Record Systems.* E1769-95. West Conshohocken, PA: ASTM; 1995.

ASTM. *Standard Guide for Confidentiality, Privacy, Access, and Data Security Principles for Health Information Including Computer-Based Patient Records.* E1869-97. West Conshohocken, PA: ASTM; 1997.

ASTM. *Standard Guide for Management of the Confidentiality and Security of Dictation, Transcription, and Transcribed Health Records.* E1902-97. West Conshohocken, PA: ASTM; 1997.

ASTM. *Standard Guide for Requests for Proposals Regarding Medical Transcription Services for Healthcare Institutions.* E1959-98. West Conshohocken, PA: ASTM; 1998.

ASTM. *Standard Guide for User Authentication and Authorization.* E1985-98. West Conshohocken, PA: ASTM; 1995.

ASTM. *Standard Guide for Information Access Privileges to Health Information.* E1986-98. West Conshohocken, PA: ASTM; 1998.

ASTM. *Standard Guide for Individual Rights Regarding Health Information.* E1987-98. West Conshohocken, PA: ASTM; 1998.

ASTM. *Standard Guide for Training of Persons Who Have Access to Health Information.* E1988-98. West Conshohocken, PA: ASTM; 1998.

ASTM. *Standard Guide for Amendments to Health Information.* E2017-99. West Conshohocken, PA: ASTM; 1999.

ASTM. *Standard Specification for Authentication of Healthcare Information Using Digital Signatures.* E2084-00. West Conshohocken, PA: ASTM; 2000.

ASTM. *Standard Guide on Security Framework for Healthcare Information.* E2085-00a. West Conshohocken, PA: ASTM; 2000.

ASTM. *Standard Guide for Internet and Intranet Healthcare Security.* E2086-00. West Conshohocken, PA: ASTM; 2000.

ASTM. *Standard Specification for Quality Indicators for Controlled Health Vocabularies.* E2087-00. West Conshohocken, PA: ASTM; 2000.

ASTM. *Standard Guide for Identification and Establishment of a Quality Assurance Program for Medical Transcription.* E2117-00. West Conshohocken, PA: ASTM; 2000.

ASTM. *Standard Guide for Coordination of Clinical Laboratory Services Within the Electronic Health Record Environment and Networked Architectures.* E2118-00. West Conshohocken, PA: ASTM; 2000.

ASTM. *Standard Practice for Modeling in Health Informatics.* E2145-01. West Conshohocken, PA: ASTM; 2001.

ASTM. *Standard Specification for Audit and Disclosure Logs for Use in Health Information Systems.* E2147-01. West Conshohocken, PA: ASTM; 2001.

ASTM. *Provisional Standard Specification for Audit and Disclosure Logs for Use in Health Information Systems.* PS115-99. West Conshohocken, PA: ASTM; 1999.

ASTM. *Standard Guide for Selection of a Clinical Laboratory Information Management System.* E792-02. West Conshohocken, PA: ASTM; 2002.

ASTM. *Standard Guide for Electronic Authentication of Health Care Information.* E1762-95. West Conshohocken, PA: ASTM; 1995.

ASTM. *Standard Specification for Transferring Information Between Clinical Instruments and Computer Systems.* E1394-97. West Conshohocken, PA: ASTM; 1997.

Beizer B. *Software Testing Techniques.* 2nd ed. New York, NY: Van Nostrand Reinhold; 1990.

Bender R. *Writing Testable Requirements.* Version 1.0. Larkspur, CA: Bender & Associates, Inc.; 1996.

Dumas J. *Designing User Interfaces for Software*. Englewood Cliffs, NJ: Prentice Hall; 1988.

Food and Drug Administration. *General Principles of Software Validation; Final Guidance for Industry and FDA Staff.* FDA; 2002.

Food and Drug Administration. *Draft Guidelines for the Validation of Blood Establishment Computer Systems*. Rockville, MD: Center for Biologics Evaluation and Research, FDA; 1993.

Hetzel B. *The Complete Guide to Software Testing.* 2nd ed. New York, NY: John Wiley & Sons, Inc.; 1993.

Horton W. *Designing and Writing Online Documentation*. New York: John Wiley and Sons, Inc.; 1990:33;151-156;164.

Kit E. *Software Testing in the Real World*. Boston, MA: Addison-Wesley Longman; 1995.

Kaner C, et al. *Testing Computer Software.* 2nd ed. New York, NY: John Wiley and Sons; 1999.

Mayhew DJ. *The Usability Engineering Lifecycle: A Practitioner's Handbook for User Interface Design.* San Francisco, CA: Morgan Kaufmann Publishers; 1999.

Norman DA, Draper SW, eds. *User-Centered Systems Design: New Perspectives on Human-Computer Interaction*. Hillside, NJ: Laurence Erlbaun; 1986.

ISO/IEC 14598. *Medical Devices - Risk Management – Part 1: Application of Risk Analysis*. International Organization for Standardization; 1998.

Rath and Strong's Six Sigma Pocket Guide. Lexington, MA: Rath and Strong; 2000.

Steane S, et al. *Guidelines for Preparing Standard Operating Procedures for Blood Bank Computer System.* Arlington, VA: American Association of Blood Banks; 1991.

Wiegers KE. *Software Requirements*. Redmond, WA: Microsoft Press; 1999.

Title 21 Code of Federal Regulations (21 CFR Part 11). Electronic Records; Electronic Signatures. Final Rule; March 1997.

Title 21 Code of Federal Regulations (21 CFR Part 11). Electronic Records; Electronic Signatures; Validation. Draft Guidance; August 2001.

Title 21 Code of Federal Regulations (21 CFR Part 11). Electronic Records; Electronic Signatures; Glossary of Terms. Draft Guidance; August 2001.

Title 21 Code of Federal Regulations (21 CFR Part 11). Electronic Records; Electronic Signatures Time Stamp. Draft Guidance; February 2002.

Title 45 of Federal Regulations (45 CFR Parts 160 and 164). Standards for Privacy of Individually Identifiable Health Information. Proposed Rule; March 2002.

Appendix A. Format of a Software Specification Document

A1. Objectives

Describe the intended use for this feature or modification and why it is important to implement the function as designed.

A2. General Functions

Include areas to be addressed within the laboratory, as well as a brief discussion about proposed features (e.g., order entry, bar-code generation, specimen identification, communications, word processing).

A3. Specific Requirements

Include details about areas that require particular attention (i.e., screen layouts, designs, [see Section A5, "Attachments"], hardware, operating system, data interchange formats, file system, communication standards/protocols, compatibility, and performance [timing and response]).

A4. Schedule

For receipt of validation completion, and targeted installation, propose dates with flexible ranges.

A5. Attachments

Attachments include screens, desired outputs, listings, or other information that indicates a desired look, report, or result.

Appendix B. Data Entry Validation Form

TEST NUMBER:__________________ DATE:______DATE ENTERED:___________

Patient identification:

Unit identification:

Processes performed:

Testing performed by:

Appendix C. Sample Test Case Report Format

Prepared by:____ Date:____________________

Description of function to be tested:

Program/module/environment:

Critical points:

Warnings:

Overrides:

Access points:

Instructions for testing:

Number of repetitions:

Documentation methods:

Acceptance criteria:

Appendix C. (Continued)

Test plan approved by:______________________ Date:______________________________

Result of Testing:

Testing by:____ Date(s):______________

Location of documentation:

Unexpected outcomes:

_____________________Criteria met _____________________Criteria not met

Corrective actions:

Retest results:

Final approval for implementation:

Approved by:____ Date:_______________________

Appendix D. Sample Test Case #1 – Expired Crossmatch

Test Number:____________________

Change: Correct dispense date and time.

Problem: Multiple expired cross-matched units were dispensed in the same conversation, or multiple outdated products were dispensed in the same conversation. The specimen date was recorded as the dispense date/time.

Test environment: Testing to be performed in the certification environment.

Critical Points:

1. Patient/unit identification
2. Dispense date/time

Access Points:

Barcode and manual keyboard entry of data

Instructions for Testing:

1. Dispense 3 units with expired cross-matches in the same conversation.
2. Dispense 3 units that were outdated at the time of dispense.
3. Determine dispense date/time.

Patient	Units	Outdate of Unit	Dispense Date/Time

Documentation Method:

Grid of data input elements and print screens of output.

Acceptance Criteria:

1. Correct patient identification tracking.
2. Correct dispense date/time.
3. No changes in dispense or routine units.

Approved by:____ Date:_____________________

Appendix D. (Continued)

Testing Results:

_______Acceptable _______ Not acceptable Reason______________________

Location of Documentation:

Box 34 B

Unexpected outcomes:

_____________________Criteria met _____________________Criteria not met

Corrective actions:

Reviewed by:____ Date:_______________________

Appendix E. Sample Test Case #2 - ABO/Rh Compatibility Table Review

Purpose:

Review the table to determine that the correct decision to allow, warn, or block activity has/will occur for each patient ABO/Rh type matched with each red blood cell or whole blood component.

Environment: Production

Critical Points:

Warn when Rh incompatible units are issued, allow dispense.
Block issue of ABO incompatible units, no override capability.

Access points: Computer printout of table structure

Instructions for Testing:

1. Print out the table (group/type compare).
2. Review the table parameters for allow, warn, or block (fields X, Y, Z).
3. Identify any incorrect or questionable entries by circling them on the report.

Number of repetitions: One - review of table structure.

Documentation Method:

Grid of data input elements and print screens of output.

Acceptance Criteria:

Whole blood may be issued only to a patient of that type.

RBC Component	Compatible Allowed	Allowed With Warning	Blocked With Warning
O neg	UKN, All		
O pos	All pos	UKN, all neg	
A neg	All AB, A		All B, O
A pos	AB & A Pos	A neg, AB neg	All B, O

ETC.

Testing performed by: ____________________ Date:____________________________

Location of documentation:

Box 6C

Testing Results:

Appendix E. (Continued)

Unexpected outcomes:

____________________Criteria met ____________________Criteria not met

Retested by:_______________________________Date:____________________________

Results:

Conclusion: _____________ Acceptable _____________ Not Acceptable

Final Review by:_________________________________ Date:______________________

Appendix F. Sample Test Case #3 – ABO/Rh Compatibility Test

Purpose:

Determine if AB Pos red blood cells (RCAS), when dispensed to patients of all other ABO/Rh types and a patient without a blood type on file, will result in the correct decision to allow, warn, or block activity.

Environment: Production

Critical Points:

Warn when Rh incompatible units are issued, allow dispense
Block issue of ABO incompatible units, no override capability.

Access points: Barcode and keyboard manual entry of data

Instructions for testing:

1. Order a cross-match for nine patients, one of each type and for the patient without a type on file.
2. Call the cross-match program.
3. Enter the accession number for the first patient.
4. Read the unit barcode, M12345 (AB pos RCAS), from the unit sheet.
5. Record the result: match allowed, warning given, or match blocked.

Number of tests to be performed:

One per patient/AB Rh combination because all combinations are tested. This function is dependent on the table settings for each product and all combinations are being tested. **Note**: This test should be repeated for **EACH** ABO/Rh type of whole blood or red blood cell component.

Documentation Method:

Grid of data input elements and print screens of output.

Acceptance Criteria:

Match allowed for AB pos patient.
Warning given when unit assigned to AB neg.
Match blocked for all other patient types and untyped patient.

Testing performed by:____ Date:______________________

Location of Documentation:

Box 45C

Appendix F. (Continued)

Testing results:

Patient Type	Accession	Allowed	Warn	Blocked
O pos	1	N	N	Y
O neg	2	N	N	Y
A pos	3	N	N	Y
A neg	4	N	N	Y
B pos	5	N	N	Y
B neg	6	N	N	Y
AB pos	7	Y	N	N
AB neg	8	N	Y	N
Untyped	9	N	N	Y

Conclusion: ______ Acceptable _____ Not Acceptable
Reason _____________

Unexpected Outcomes:

Corrective action:

Retested by:____ Date:______________________

Result:

Conclusion: _______ Acceptable _______ Not Acceptable
Reason______________

Final review by:____ Date:______________________

Conclusion:

Appendix G. Sample Test Case #4 – Patient’s Historical Blood Type

Purpose:

Establish confidence that the patient's ABO and Rh type on file are the same as the conclusions of the typing results.

Environment: Production

Critical Points:

Blood Types of historical file and typing result conclusions match.

Access points: Historical file and patient result files

Instructions for Testing:

1. Run the program to compare historical blood type with blood type conclusion in the computer of individual typing tests.

2. Review printout of cases where blood types are discrepant.

Number of tests to be performed:

All results from 2/14/98 to 2/15/02 (comparison of over 10,000 results).

Acceptance Criteria:

All discrepancies explained. Acceptable reagents for discrepancies include bone marrow transplant, newborn type, massive transfusion, mislabeled specimen previously submitted, and insurance fraud.

Testing performed by:____ **Date**:___________________

Testing results:

Location of Documentation

Box 3D

Conclusion:_____________________________________ **Acceptable** _________ **Not Acceptable**
Reason____________________

Unexpected Outcomes:

Corrective action:

Retested by:____ **Date**:_____________________

Results:

Conclusion:_____________________________________ **Acceptable** _________ **Not Acceptable**
Reason____________________

Final review by:____ **Date**:_____________________

Conclusion:

Appendix H. Hospital Information System (HIS) Interface Validation

Purpose:

To validate that laboratory results are transmitted and reported accurately, check the interface between the laboratory information and the hospital information systems periodically.

General Information:

This procedure describes what results to review, as well as how and where to document the information. Results from tests performed the previous day on three patients are compared in the laboratory database with those in the hospital systems. In addition, "stress" results are entered on a quality assurance patient to validate that the unusual types of data are being transmitted. Validation is performed quarterly and after installation of software for the interfaces, for example, MNOBxxxx or HLxxxM.

Environment: Certification

Critical Points:

Data is transmitted as expected.

Access points:

Data transmitted from LIS to HIS

Instructions for Testing

1. Validated clinical laboratory information system results from previous day are transmitted to hospital information systems.

   Print cumulative summary and database reports, including pending, canceled, and verified tests from KDS on one patient from each of the following locations:

   a. 4A
   b. 4D
   c. MED

   Select patients with blood bank, microbiology, chemistry, and hematology tests.

Documentation Method:

a. Compare the results from cumulative summary and database query with those on the hospital systems.

b. If results are identical in the other systems, record "OK" or place a check mark in the appropriate box on the HIS Interface Quality Assurance Documentation form. Save the printout in the current month's completed Users' Services file.

c. If the results are not acceptable in the other systems, record "No" on the HIS Interface Quality Assurance Documentation Form, troubleshoot the problem, and notify LIS supervisors for quality assurance review and resolution. Record action taken on the back of the form.

Appendix H. (Continued)

APPROVED BY:	REVIEWED BY: DATE:
APPROVED DATE:	REVIEWED BY: DATE:
EFFECTIVE: THRU:	REVIEWED BY: DATE:

2. Enter "stress" results in database on a Quality Assurance patient (00000034) and validate that they are being transmitted to hospital systems.

   a. Use "Order Entry" to order the test on patient 00000034, specifying "N" to the question "Collected?"

   b. Check the hospital systems to validate that the status is as follows:

      H1-Pending
      H2-Ordered

      Record "OK" or place a check mark in the appropriate box on the HIS Interface Quality Assurance Documentation Form if the status is correct. If not, record "No" and the action taken on the back of the form.

   c. Use "Specimen Receipt by Specimen" to receive the specimen.

   d. Check the hospital systems to validate that the status is as follows:

      H1-Pending
      H2-In Lab

      Record "OK" or place a check mark in the appropriate box on the HIS Interface Quality Assurance Documentation Form if the status is correct. If not, record "No" and the action taken on the back of the form.

   e. Add the test to worksheet (routine urinalysis master).

   f. Enter preliminary results and verify the following results:

      1. PH - 9.5 (Normal range = 4.5-8.0)

      2. BILI = POS (Normal = NEG)

      3. Enter a total of 34 lines of comments at the sequence and test levels.

   g. Check the hospital systems to validate the status is as follows:
      H1-Result + comment "Prelim."
      H2-Result + a "P."
      H3-Comment indicating that the test is Preliminary.

      Record "OK" or place a check mark in the appropriate box on the HIS Quality Assurance Documentation Form if the status is acceptable. If not, record "No" and the action taken on the back of the form.

Appendix H. (Continued)

h. Enter final results and verify the results.

i. Print a cumulative summary report on the quality assurance patient.

j. Review the test results from the quality assurance patient on hospital systems.

k. If results are identical in the other systems, record "OK" or place a check mark in the appropriate box on the HIS Quality Assurance Documentation Form. If not, record "No" and the action taken on the back of the form.

l. If any test or result is not acceptable in the other systems, troubleshoot the problem and notify the LIS supervisors for quality assurance review and resolution.

m. Use "Test Delete with Results" to delete a test.

n. Check the hospital systems to validate that the status is as follows:

H1-Canceled
H2-Canceled
H3-Canceled

Record "OK" or place a check mark in the appropriate box on the HIS Quality Assurance Documentation Form if the status is acceptable. If not, record "No" and the action taken on the back of the form.

3. Validate that results are transmitted to hospital systems. (This includes Surgical Pathology, Special Hematology, and Cytology Departments.)

a. Print one example of a Surgical Pathology, Special Hematology, and Cytology database report from patients at each of the following locations:

1. 4A
2. 4D

b. Compare the results from KDS with those on the hospital information systems.

c. If results are identical in the other systems, record "OK" or place a check mark in the appropriate box on the HIS Interface Quality Assurance Documentation Form. If not, record "No" and the action taken on the back of the form.

d. If the status or results are not acceptable in the other systems, troubleshoot the problem and notify LIS supervisors for quality assurance review and resolution.

4. Staple all database printouts to the HIS Interface Quality Assurance Documentation Form and save them in the current month's completed Users' Services file.

Appendix I. HIS Interface Quality Assurance Documentation

	H1		H2		H3	
Three Patients	Acceptable*	Initials/Date	Acceptable*	Initials/Date	Acceptable*	Initials/Date
Numeric						
ASCII						
Comment						
Pending Relt						
Cancel Relt						
Grm Stn Relt						
Culture Relt						
Sensitivity						
BB Unit Avail						
BB Usage						
Sur Path Relt						
Home Relt						
Cytol Relt						
QA PAT						
Order						
Rec'd						
Prelim ASCII						
Prelim Num						
Prelim Com't						
Final ASCII						
Final Num						
Final Com't						

*Acceptable: Record "OK" or place a check mark in the box if the test appears acceptable in the appropriate system. If not, record "No," and record the action taken on the back of the form.

Abbreviations:

Num = Numeric.	Sur Path = Surgical pathology.
Relt = Result.	Cytol = Cytology.
Grm Stn = Gram stain.	QA Pat = Quality assurance patient.
BB Unit Avail = Blood bank unit cross-matched/set-up.	Rec'd = Received.
BB Usage = Blood bank units transfused.	Prelim = Preliminary.
	Com't = Comment.

Appendix I. (Continued)

Acceptance Criteria:

No discrepancies found.

Testing results:

Location of Documentation

Box 3D

Conclusion: ________ **Acceptable** ________ **Not Acceptable**
Reason____________________

Unexpected Outcomes:

Corrective action:

Retested by:____ **Date**:_____________________

Results:

Conclusion:____________________________________ **Acceptable** _________ **Not Acceptable**
Reason____________________

Final review by:_______________________________________ **Date:**_____________________

Appendix J. Worksheet Calculations Validation

Purpose:

The laboratory-accrediting agency requires that calculations performed by the laboratory information system on worksheets be validated periodically.

General Information:

Make validation of the information system's calculations coincide with the annual review of procedures. The supervisor of each laboratory area is responsible for completing the validation for worksheets in their area.

Environment: Production

Critical Points: Calculations are performed as programmed.

Access points: Review of actual data entered and manual calculations vs. computer calculations.

Instructions for Testing:

1. Print the worksheet definition for each worksheet containing a calculation.
2. Print a sequence from each worksheet containing a calculation.
3. On the worksheet definition, record near the calculation section the value(s) for each item used in a calculation and perform the calculation.
4. Record the results of manual calculations on the worksheet definition near the calculation section.
5. If the manual and computer results are identical, indicate that the calculations are acceptable, and sign and date the form.
6. If results are not acceptable, investigate the problem and record the action taken.
7. Send all documentation to LIS and Users' Services.

Number of Repetitions:

Once for each calculation made by the computer system.

Appendix J. (Continued)

Documentation Method:

Complete worksheet and grid for each test.

Test Name	**Manual Calculation**	**Computer Results**	**Acceptable** Y or N

Acceptance Criteria

Manual and automated calculations agree.

Testing results:

Location of Documentation

Box 3D

Conclusion: _________ **Acceptable** _________ **Not Acceptable** **Reason**____________________

Unexpected Outcomes:

Corrective action:

Retested by:____ **Date**:_____________________

Results:

Conclusion:_________ **Acceptable** _________ **Not Acceptable** **Reason**____________________

Final review by:____

Appendix K. Example of Validation of Worksheet Calculations

```
WORKSHEET OCBCPD              RUN #        1 STKS WORKSHEET

                                                                  12/17/92
1      1496018     DIAG:      V76.9        SCREEN-NEOPLASM NOS

                   J                                              042 M

12/17/92 07:30

       Add                                         pr/et          prior et
ODFA:        ODFA                WBC                7.2           RBC    5.15
             raw%          raw#   report%          report#        HGB    14.9
       NE    61.2          4.3    61.2             4.3            HCT    44.5
       LY    24.6          1.8    24.6             1.8            MCV    86
       NO    12.0          .9     12.0             0.9            MCH    29.0
       ED    1.1           .1     1.1              0.1            MCHC   33.6
       BA    1.1           .1     1.1              0.1            RDW    12.8

REF RNG
 NEUT        42.0000       70.0000                           PLT  244
 LYMPH       20.0000       48.0000                           MPV  8.2


2      1503847     Diag:  V42.1      HEART TRANSPLANT STATUS

                   C                                              O48 F

12/17/92 07:01

       Add                                         pr/et          prior et
ODFA:        ODFA                WBC                10.0          RBC    3.83
             raw%          raw#   report%          report#        HGB    13.3
       NE    64.7          8.5    ******           ******         HCT    39.2
       LY    9.2           .9     ******           ******         MCV    102*
       NO    3.9           .4     ******           ******         MCH    34.7*
       ED    .2            .0     ******           ******         MCHC   33.9
       BA    2.0           .2     ******           ******         RDW    27.2*

REF RNG
 NEUT        42.0000       70.0000                           PLT  174
 LYMPH       20.0000       48.0000                           MPV  8.3
```

Appendix K. (Continued)

```
 3       1503904       Diag:   V42.1          HEART TRANSPLANT STATUS

                                                                  O20 M
12/17/92 07:03

        Add                                    pr/et          prior et
ODFA:           ODFA             WBC           10.0           RBC    2.95*

                raw%            raw#   report%       report#        HGB    9.4*
        NE      88.8            11.4   ******        ******         HCT    21.1*
        LY      4.6             .6     ******        ******         MCV    92
        NO      5.2             .9     ******        ******         MCH    31.7
        ED      .4              .1     ******        ******         MCHC   34.6
        BA      .0              .0     ******        ******         RDW    29.5*

REF RNG
 NEUT                   42.0000               70.0000               PLT    185
 LYMPH                  20.0000               48.0000               MPV    7.1
```

Appendix K. (Continued)

WORKSHEET DEFINITION REPORT DATE 12/15/92 TIME 9:41 PAGE: 1

WORKSHEET ID: OCHCPD NAME: STMB WORKSHEET
Status: C

PARAMETERS

Run Size:	800
Page Length:	60
Sort Order:	N
Reset:	Y
Priority:	N

Applying Rules: N Delta Warning: Y Default Status: F

Page Header ID:	OCBCPD
Worksheet Location	
Facility:	DM
Lab:	OP
Lab Section:	OPD
Work Station:	

RESERVE LIST

Sequence #	Format	Known Type	Known ID

SCREEN #1 TEMPLATE #9

Add				pr/et	prior et	
ODFA:	_______	WBC		________	RBC	_______
	raw%	raw#	report%	report#	HGB	_______
NE	_______	_______	******	******	HCT	_______
LY	_______	_______	******	******	MCV	_______
NO	_______	_______	******	******	MCH	_______
ED	_______	_______	******	******	MCHC	_______
BA	_______	_______	******	******	RDW	_______

REF RNG
NEUT PLT
LYMPH MPV

LINE #	#	ITEM	ITEM DESCRIPTION
2	1	SEQ#	
2	2	SPEC#	
2	3	DIAG	
4	4	NAME	
4	5	PT#	
4	6	AGE	
4	7	SEX	
	8	COLL	

Appendix K. (Continued)

WORKSHEET DEFINITION REPORT DATE 12/15/92 TIME 9:41 PAGE: 2

6	9	REQ-COM	UM OP ODFA		B	OPSTD	
8	10						
	11		UM OP OW		B	OPSI	
	12	PRIORET	UM OP OW		B	OPSI	
8	13		UM OP OR		B	OPSI	
8	14	PRIOR	UM OP OR		B	OPSI	
8	15	ET	UM OP OR		B	OPSI	
9	16		UM OP OH		B	OPSTD	
9	17	PRIOR	UM OP OH		B	OPSTD	
9	18	CT	UM OP OH		B	OPSTD	
10	19						
10	20						
10	21		UM OP	10	B	OPSTD	ONEU Y
10	22		UM OP	10	B	OPSTD	ON AB Y
10	23	PRIORET	UM OP ODFA		B	OPSTD	ONEU
10	24		UM OP OHT		B	OPSTD	
10	25	PRIOR	UM OP OHT		B	OPSTD	
10	26	ET	UM OP OHT		B	OPSTD	
11	27						
11	28						
11	29		UM OP	10	B	OPSTD	OLYMPH Y
11	30		UM OP	10	B	OPSTD	OL ABS Y
11	31	PRIORET	UM OP ODFA		B	OPSTD	OLYMPH
11	32		UM OP OMCV		B	OPSI	
11	33	PRIOR	UM OP OMCV		B	OPSI	
11	34	CT	UM OP OMCV		B	OPSI	
12	35						
12	36						
	37		UM OP	10	B	OPSTD	OMONO Y
12	38		UM OP	10	B	OPSTD	OMAB Y
12	39	PRIORET	UM OP ODFA		B	OPSTD	OMONO Y
12	40		UM OP OMCH		B	OPSI	
12	41	PRIOR	UM OP OMCH		B	OPSI	
12	42	ET	UM OP OMCH		B	OPSI	
13	43						
13	44						
13	45		UP OP	10	B	OPSTD	OEOS
13	46		UM OP	10	B	OPSTD	OEAB
13	47	PRIORET	UM OP ODFA		B	OPSTD	OEOS
13	48		UM OP OMCHC		B	OPSTD	
13	49	PRIOR	UM OP OMCHC		B	OPSTD	
13	50	ET	UM OP OMCHC		B	OPSTD	
14	51						
14	52						
14	53		UM OP	10	B	OPSTD	OBAS Y
14	54		UM OP	10	B	OPSTD	OBAB Y
14	55	PRIORET	UM OP ODFA		B	OPSTD	OBAS
14	56		UM OP ORDW		B	OPSTD	

Appendix K. (Continued)

WORKSHEET DEFINITION REPORT DATE 12/15/92 TIME 9:41 PAGE: 3

14	57	PRIOR	UM OP ORDW	B	OPSTD	
14	56		UM OP ORDW	B	OPSTD	
14	57	PRIOR	UM OP ORDW	B	OPSTD	
14	58	CT	UM OP ORDW	B	OPSTD	
16	59	NORM-LOW	UM OP ODFA	B	OPSTD	ONEU
16	60	NORM-HIGH	UM OP ODFA	B	OPSTD	ONEU
16	61		UM OP OPLT	B	OPSI	
16	62	PRIOR	UM OP OPLT	B	OPSI	
16	63	ET	UM OP OPLT	B	OPSI	
17	64	NORM-LOW	UM OP ODFA	B	OPSTD	OLYMPH
17	65	NORM-HIGH	UM OP ODFA	B	OPSTD	OLYMPH
17	66		UM OP OMPV	B	OPSI	
17	67	PRIOR	UM OP OMPV	B	OPSI	
17	68	ET	UM OP OMPV	B	OPSI	

Equation for Item # 21 =
(K1*F19) 1 x 61.2 = 61.2

Equation for Item # 22 =
(K1*F20) 1 x 4.3 = 4.3

Equation for Item # 29 =
(K1*F27) 1 x 24.6 = 24.6

Equation for Item # 30 =
(K1*F28) 1 x 1.8 = 1.8

Equation for Item # 37 =
(K1*F35) 1 x 12.0 = 12.0

Equation for Item # 38 =
(K1*F36) 1 x 0.9 =

Equation for Item # 45 =
(K1*F43) 1 x 1.1 = 1.1

Equation for Item # 46 =
(K1*F44) 1 x 0.1 = 0.1

Equation for Item # 53 =
(K1*F51) 1 x 1.1 = 1.1

Equation for Item # 54 =
(K1*F52) 1 x 0.1 = 0.1

Appendix K. (Continued)

Initialize Alanine Aminotransferase (ALT) Transfer on ALT Batch Processor

Purpose:

This test plan is intended to ensure that ALT controls are received and recorded by the laboratory computer but not evaluated for validity; that the maximum number of samples between controls is set at 58; and that a batch transferred to the laboratory information system contains no more than five control sets.

Program: ALT control transfer in certification environment

Critical Points:

Data transfer is without error.

Access points: Transfer of data from instrument to laboratory computer systems.

Number of repetitions:
Two sets of transfer data

Documentation Method

On test grid

Test name	Number Transferred	Number Received	Test Results OK Y or N

Instructions for Testing:

A. Date Setup

1. Unless otherwise indicated, controls are within ±2 SD of the mean.

2. Create results for ALT instrument as follows:

   a. Batch with both beginning controls, 58 samples, and both ending controls.

   b. Batch with only both beginning and both ending controls; no donor samples.

   c. Batch with beginning abnormal control, no beginning normal control, samples, and both ending controls.

   d. Batch with both beginning controls, 59 samples, both ending controls.

   e. Batch with beginning controls, samples, normal followed by abnormal controls, samples, both ending controls.

Appendix K. (Continued)

3. Manually create results for ALT instrument. Enter both beginning controls ±2 SD, samples and ending controls as listed below:

	Controls		Batch Interpretation
	Abnormal	Normal	
a.	< 2 SD	< 2 SD	Valid
b.	> 3 SD	> 3 SD	Valid

4. Manually create results for ALT Instrument. Enter all controls within ± 2 SD.

   a. Beginning controls, 56 samples, controls, 58 samples, controls, 58 samples, controls, 58 samples, and ending controls.

   b. Beginning control, 56 samples, controls, 58 samples, controls, 58 samples, controls, 58 samples, and ending controls.

B. Successful Operations

1. Transfer results from the ALT instrument as follows:

   a. Batch with both beginning controls, 58 samples, and both ending controls.

      ALT results are accepted. The report indicates that the batch was processed and was acceptable. There is no reference to the validity status of any control.

      Initials:______________ Date:________________

   b. Batch with only both beginning and both ending controls; no donor samples.

      ALT results are accepted. The report indicates that the batch was processed and was acceptable. There is no reference to the validity status of any control.

      Initials:______________ Date:________________

   c. Batch with beginning abnormal control, no beginning normal control, samples, and both ending controls.

      ALT results are not accepted. The report indicates that the batch was not processed.

      Error message: Unexpected Control ID:

      Initials:______________ Date:________________

   d. Batch with both beginning controls, 59 samples, both ending controls.

Appendix K. (Continued)

ALT results are not accepted. The report indicates that the batch was not processed.

Error message: Control Interval limit (58) exceeded at sample number nn

Initials: _____________ Date:_______________

e. Batch with beginning controls, samples, normal followed by abnormal controls, samples, both ending controls.

ALT results are not accepted. Report indicates that the batch was not processed.

Error message: "Unexpected Control ID. Maximum number of controls exceeded. Batch invalid."

Initials: ____________________ Date:______________________

2. Transfer results from ALT instrument. Enter both beginning controls as listed below:

	Controls		
	Abnormal	**Normal**	**Batch Interpretation**
a.	< 2 SD	< 2 SD	Valid
b.	> 3 SD	> 3 SD	Valid

3. Acceptance Criteria:

ALT results are accepted. The report indicates that the batch was processed and was acceptable. There is no reference to the validity status of any control.

Initials:___________________________ Date:_________

4. Transfer results from ALT instrument. Enter all controls within ±2 SD.

a. Beginning controls, 50 samples, controls, 50 samples, controls, 58 samples, controls, 58 samples, and ending controls.

ALT results are accepted. The report indicates that the batch was processed and was acceptable. There is no reference to the validity status of any control.

Initials:___________________________ Date:_________

b. Beginning controls, 56 samples, controls, 58 samples, controls, 58 samples, controls, 58 samples, controls, 58 samples, and ending controls.

ALT results are accepted. The report indicates that the batch was not processed.

Error message: "Maximum number of controls exceeded. Batch invalid."

Initials:___________________________ Date:_________

Appendix K. (Continued)

Testing results:

Location of Documentation: Box 3D

Conclusion: **Acceptable** ___________ **Not Acceptable** ____________**Reason**___________________

Unexpected Outcomes:

Corrective action:

Retested by:____ **Date**:_____________________

Results:

Conclusion: **Acceptable** ___________ **Not Acceptable** ____________**Reason**___________________

Final review by:___________________

NCCLS consensus procedures include an appeals process that is described in detail in Section 9 of the Administrative Procedures. For further information contact the Executive Offices or visit our website at www.nccls.org.

Summary of Comments and Area Committee Responses

GP19-A: *Laboratory Instruments and Data Management Systems: Design of Software User Interfaces and End-User Software Systems Validation, Operation, and Monitoring; Approved Guideline*

General

1. GP19-A tries to cover too many kinds of software in too many phases of development and installation. Two questions can summarize my concerns:

   1. What should apply to **development** and what to **testing** of a "finished product"?
   2. What testing should apply to a new element of an instrument and an entire LIS, respectively?

   Both of these areas are mentioned in the "Summary of Comments and Subcommittee Responses." (Numbers 2, 5, 7, 8, for example).

   Confusion over what should be applied could result in laboratories encountering enormous costs trying to live up to unrealistic expectations. For example, the kind of "Worksheet Calculations" shown in Appendix I (now Appendix K) may be appropriate to testing a new instrument, but if applied to all instruments connected to a new LIS could greatly inflate the cost of LIS testing without any real benefit. An LIS deals with the digital result from an instrument, not the transducers and analog elements within the instrument.

- **Additional text has been added to the Introduction of this document to describe the rationale of the area committee to maintain this document as it is currently structured.**

  Additional text has also been added to Section 11.2 stating that the vendor will provide a test plan, including documentation necessary to allow the end user to validate the user's system, as well as items required to validate in the end user setting. This plan should include a particular focus on end-user-defined procedures.

2. I did not see addressed the issue of power on reset. Items to be addressed in this area include: establishing a safe or nonfaulted state on power-up or loss of power; determining the maximum allowable cycle time when the machine is reset (i.e., if the cycle time is too long the instrument may not meet the user's needs); and whether reset is soft (hot) versus hard (cold).

- **The issue of power on reset should be addressed by the manufacturer and is beyond the scope of this document.**

Section 4.2.2.1

3. A system should limit short-term memory requirements associated with tasks. It is really the minimization of judgment associated with tasks. I don't think that we are dealing with SIMULTANEOUS tasks. We are referring to repetitive tasks, or duplication of effort within a process. I think that having to re-enter the ID# on each screen is more the issue. The sample ID is not the patient ID#, so it is necessary to tie the patient ID to the sample. That becomes a significant working memory task. Along the same lines, if a piece of information is required that one would have to look up, that should be called up into a specified location of orders previously placed within two hours.

- **The text has been modified with the addition of the following sentence: "It should not be necessary to remember the special context of information, for example: The specimen and patient identifiers should be associated so one identifier can be accessed through the other."**

Sections 4.2.2.2 and 4.2.2.3

4. An example would be automatic display of previous results when a test is verified so that the delta check is active and without technologist intervention. One can also automate the Hb or platelet count display when an order is accessioned for a cross-match. Similarly, a request for units for a surgical procedure should display the ICD code and allowed units by the maximum surgical blood order schedule.

- **The text has been modified with the addition of: "An example would be automatic display of previous results when a test is verified so that the delta check is active and without technologist intervention."**

Section 4.2.2.5

5. Computers can also be used when decisions are based on rules derived from repeated experience. What tasks require decision-making ability? The point is to reduce this factor.

- **The area committee agrees with this comment; however, this subject is beyond the scope of this document. The primary focus of this document is the software user interface within the centralized laboratory environment.**

Section 4.4

6. Multiattribute retrieval can also be semiautomated through a pattern recognition function using rules. Learning can be incorporated using Eugene Rypka's principles of truth table comprehension and is very important for quality assurance of the data generated.

- **The text and Table 2 have been modified with the addition of: "With increasing capabilities of the systems the pattern recognition, multiattribute retrieval, and learning tasks can be allocated to the computer system, but it is still the user's responsibility to validate these tasks."**

7. As a special requirement the design section could address the option of disabling any unused keyboard functions, or to put it another way, only enable those key functions that are expected to be operated at any given time. By disabling keyboard functions one should be able to write fewer error-handling routines.

- **To address the issue regarding keyboard function, the following statement has been added to Section 7.2.1: "Similar principles should be applied to keyboard functions."**

Section 9

8. This discussion is thorough, but somewhat dated. There is insufficient attention to error by an elimination algorithm.

- **Error elimination is the responsibility of the vendor and is very closely related to the products. The scope of this document deals with the presentation of errors and user interaction-induced errors.**

9. I did not see any information related to audible and visual alarms—for example, priority, alarm duration, and alarm loudness. I believe that these would all be appropriate things to be addressed by this guideline.

- **Error recognition has been addressed in Section 9.6.**

References

10. In the reference section I suggest the following: update the reference to the IEEE verification and validation plans to revision year date of 1993; also include as a reference IEEE 1012-1998, Standard for Software Verification and Validation.

- **The recommended references have been updated in the document.**

Related NCCLS Publications[*]

GP2-A4	**Clinical Laboratory Technical Procedure Manuals; Approved Guideline—Fourth Edition (2002).** This document provides guidance on development, review, approval, management, and use of policy, process, and procedure documents in the laboratory testing community.
GP18-A	**Laboratory Design; Approved Guideline (1998).** This guideline provides a foundation of information about laboratory design elements that can be used to help define the issues being considered when designing a laboratory.
GP21-A	**Training Verification for Laboratory Personnel; Approved Guideline (1995).** This document provides background and recommends an infrastructure for developing a training verification program that meets quality/regulatory objectives.

[*] Proposed- and tentative-level documents are being advanced through the NCCLS consensus process; therefore, readers should refer to the most recent editions.

Active Membership (as of 1 January 2003)

Sustaining Members

Abbott Laboratories
American Association for Clinical Chemistry
Beckman Coulter, Inc.
BD and Company
bioMérieux, Inc.
CLMA
College of American Pathologists
GlaxoSmithKline
Ortho-Clinical Diagnostics, Inc.
Pfizer Inc
Roche Diagnostics, Inc.

Professional Members

American Academy of Family Physicians
American Association for Clinical Chemistry
American Association for Respiratory Care
American Chemical Society
American Medical Technologists
American Society for Clinical Laboratory Science
American Society of Hematology
American Society for Microbiology
American Type Culture Collection, Inc.
Asociacion Mexicana de Bioquimica Clinica A.C.
Assn. of Public Health Laboratories
Assoc. Micro. Clinici Italiani- A.M.C.L.I.
British Society for Antimicrobial Chemotherapy
Canadian Society for Medical Laboratory Science—Société Canadienne de Science de Laboratoire Médical
Clinical Laboratory Management Association
COLA
College of American Pathologists
College of Medical Laboratory Technologists of Ontario
College of Physicians and Surgeons of Saskatchewan
ESCMID
International Council for Standardization in Haematology
International Federation of Biomedical Laboratory Science
International Federation of Clinical Chemistry
Italian Society of Clinical Biochemistry and Clinical Molecular Biology
Japan Society of Clinical Chemistry
Japanese Committee for Clinical Laboratory Standards
Joint Commission on Accreditation of Healthcare Organizations
National Academy of Clinical Biochemistry
National Association of Testing Authorities – Australia
National Society for Histotechnology, Inc.
Ontario Medical Association Quality Management Program-Laboratory Service
RCPA Quality Assurance Programs PTY Limited
Sociedade Brasileira de Analises Clinicas
Sociedad Espanola de Bioquimica Clinica y Patologia Molecular
Taiwanese Committee for Clinical Laboratory Standards (TCCLS)
Turkish Society of Microbiology

Government Members

Association of Public Health Laboratories
Armed Forces Institute of Pathology
BC Centre for Disease Control
Centers for Disease Control and Prevention
Centers for Medicare & Medicaid Services/CLIA Program
Centers for Medicare & Medicaid Services
Chinese Committee for Clinical Laboratory Standards
Commonwealth of Pennsylvania Bureau of Laboratories
Department of Veterans Affairs
Deutsches Institut für Normung (DIN)
FDA Center for Devices and Radiological Health
FDA Center for Veterinary Medicine
FDA Division of Anti-Infective Drug Products
Iowa State Hygienic Laboratory
Massachusetts Department of Public Health Laboratories
National Center of Infectious and Parasitic Diseases (Bulgaria)
National Health Laboratory Service (South Africa)
National Institute of Standards and Technology
New York State Department of Health
Ohio Department of Health
Ontario Ministry of Health
Pennsylvania Dept. of Health
Saskatchewan Health-Provincial Laboratory
Scientific Institute of Public Health; Belgium Ministry of Social Affairs, Public Health and the Environment
Swedish Institute for Infectious Disease Control

Industry Members

AB Biodisk
Abbott Laboratories
Abbott Laboratories, MediSense Products
Acrometrix Corporation
Alifax S.P.A.
Ammirati Regulatory Consulting
Anaerobe Systems
A/S ROSCO
Asséssor
AstraZeneca
AstraZeneca R & D Boston
Avant Immunotherapeutics, Inc.
Aventis
Axis-Shield POC AS
Bayer Corporation – Elkhart, IN
Bayer Corporation – Tarrytown, NY
Bayer Corporation – West Haven, CT
Bayer Medical Ltd.
BD
BD Biosciences – San Jose, CA
BD Consumer Products
BD Diagnostic Systems
BD Italia S.P.A.
BD VACUTAINER Systems
Beckman Coulter, Inc.
Beckman Coulter, Inc. Primary Care Diagnostics
Beckman Coulter K.K. (Japan)
Bio-Development SRL
Bio-Inova Life Sciences International
Bio-Inova Life Sciences North America
BioMérieux (NC)
bioMérieux, Inc. (MO)
Biometrology Consultants
Bio-Rad Laboratories, Inc.
Bio-Rad Laboratories, Inc. - France
Blaine Healthcare Associates, Inc.
Bristol-Myers Squibb Company
Canadian External Quality Assessment Laboratory
Chiron Corporation
ChromaVision Medical Systems, Inc.
Chronolab Ag
Clinical Design Group Inc.
Cognigen
Community Medical Center (NJ)
Copan Diagnostics Inc.
Cosmetic Ingredient Review
Cubist Pharmaceuticals
Dade Behring Inc. - Deerfield, IL
Dade Behring Inc. - Glasgow, DE
Dade Behring Inc. - Marburg, Germany
Dade Behring Inc. - Sacramento, CA
Dade Behring Inc. - San Jose, CA
David G. Rhoads Associates, Inc.
Diagnostics Consultancy
Diagnostic Products Corporation
Eiken Chemical Company, Ltd.
Elan Pharmaceuticals
Electa Lab s.r.l.
Enterprise Analysis Corporation
Essential Therapeutics, Inc.
EXPERTech Associates, Inc.
F. Hoffman-La Roche AG
Fort Dodge Animal Health
General Hospital Vienna (Austria)
Gen-Probe
GlaxoSmithKline
Greiner Bio-One Inc.
IGEN Inc.
Immunicon Corporation
Instrumentation Laboratory
International Technidyne Corporation
IntraBiotics Pharmaceuticals, Inc.
I-STAT Corporation
Johnson and Johnson Pharmaceutical Research and Development, L.L.C.
LAB-Interlink, Inc.
Laboratory Specialists, Inc.
Labtest Diagnostica S.A.
LifeScan, Inc. (a Johnson & Johnson Company)
Lilly Research Laboratories
Macemon Consultants
Medical Device Consultants, Inc.
Merck & Company, Inc.
Minigrip/Zip-Pak
mvi Sciences (MA)
Nichols Institute Diagnostics (Div. of Quest Diagnostics, Inc.)
NimbleGen Systems, Inc.
Nissui Pharmaceutical Co., Ltd.
Nippon Becton Dickinson Co., Ltd.
Norfolk Associates, Inc.
Novartis Pharmaceuticals Corporation
Ortho-Clinical Diagnostics, Inc. (Rochester, NY)
Ortho-McNeil Pharmaceutical
Oxoid Inc.
Paratek Pharmaceuticals
Pfizer Inc
Pharmacia Corporation
Philips Medical Systems
Powers Consulting Services
Premier Inc.
Procter & Gamble Pharmaceuticals, Inc.
QSE Consulting
Quintiles, Inc.
Radiometer America, Inc.
Radiometer Medical A/S
Replidyne
Roche Diagnostics GmbH
Roche Diagnostics, Inc.
Roche Laboratories (Div. Hoffmann-La Roche Inc.)
Sarstedt, Inc.
SARL Laboratoire Carron (France)
Schering Corporation
Schleicher & Schuell, Inc.
Second Opinion
Streck Laboratories, Inc.
Synermed Diagnostic Corp.
Sysmex Corporation (Japan)
Sysmex Corporation (Long Grove, IL)
The Clinical Microbiology Institute
The Toledo Hospital (OH)
Theravance Inc.
Transasia Engineers
Trek Diagnostic Systems, Inc.
Tyco Kendall Healthcare
Versicor, Inc.
Vetoquinol S.A.
Vysis, Inc.
Wyeth-Ayerst
Xyletech Systems, Inc.
YD Consultant
YD Diagnostics (Seoul, Korea)

Trade Associations

AdvaMed
Japan Association Clinical Reagents Ind. (Tokyo, Japan)
Medical Industry Association of Australia

Associate Active Members

31st Medical Group/SGSL (APO, AE)
67th CSH Wuerzburg, GE (NY)
121st General Hospital (CA)
Academisch Ziekenhuis-VUB (Belgium)
Acadiana Medical Laboratories, LTD (LA)
Advocate Healthcare Lutheran General (IL)
Akershus Central Hospital and AFA (Norway)
Albemarle Hospital (NC)
Allegheny General Hospital (PA)
Allina Health System (MN)
Alton Ochsner Medical Foundation (LA)
Anne Arundel Medical Center (MD)
Antwerp University Hospital (Belgium)
Arkansas Department of Health
ARUP at University Hospital (UT)
Armed Forces Research Institute of Medical Science (APO, AP)
Associated Regional & University Pathologists (UT)
Atlantic Health System (NJ)
Aurora Consolidated Laboratories (WI)
AZ Sint-Jan (Belgium)
Azienda Ospedale Di Lecco (Italy)
Baxter Regional Medical Center (AR)
Bay Medical Center (MI)
Baystate Medical Center (MA)
Bbaguas Duzen Laboratories (Turkey)
Bermuda Hospitals Board
Bo Ali Hospital (Iran)
Brooks Air Force Base (TX)
Broward General Medical Center (FL)
Cadham Provincial Laboratory
Calgary Laboratory Services
Cape Breton Healthcare Complex (Nova Scotia, Canada)
Carilion Consolidated Laboratory (VA)
Cathay General Hospital (Taiwan)
Central Peninsula General Hospital (AK)
Central Texas Veterans Health Care System
Centro Diagnostico Italiano (Milano, Italy)
Champlain Valley Physicians Hospital (NY)
Chang Gung Memorial Hospital (Taiwan)
Changi General Hospital (Singapore)
The Charlotte Hungerford Hospital (CT)
Children's Hospital (LA)
Children's Hospital (NE)
Children's Hospital & Clinics (MN)
Children's Hospital Medical Center (Akron, OH)
Children's Hospital of Philadelphia (PA)
Children's Medical Center of Dallas (TX)
Clarian Health–Methodist Hospital (IN)
Clinical Laboratory Partners, LLC (CT)
CLSI Laboratories (PA)
Columbia Regional Hospital (MO)
Commonwealth of Kentucky
Community Hospital of Lancaster (PA)
CompuNet Clinical Laboratories (OH)
Cook County Hospital (IL)
Cook Children's Medical Center (TX)
Covance Central Laboratory Services (IN)
Danish Veterinary Laboratory (Denmark)
Danville Regional Medical Center (VA)
Dean Medical Center (WI)
Department of Health & Community Services (New Brunswick, Canada)
DesPeres Hospital (MO)
Detroit Health Department (MI)
Diagnosticos da América S/A (Brazil)
Dr. Everett Chalmers Hospital (New Brunswick, Canada)
Doctors Hospital (Bahamas)
Duke University Medical Center (NC)
Dwight David Eisenhower Army Med. Ctr. (GA)
E.A. Conway Medical Center (LA)

East Side Clinical Laboratory (RI)
Eastern Health (Vic., Australia)
Elyria Memorial Hospital (OH)
Emory University Hospital (GA)
Fairview-University Medical Center (MN)
Federal Medical Center (MN)
Florida Hospital East Orlando
Focus Technologies (CA)
Foothills Hospital (Calgary, AB, Canada)
Fresenius Medical Care/Spectra East (NJ)
Fresno Community Hospital and Medical Center
Frye Regional Medical Center (NC)
Gambro BCT (CO)
Geisinger Medical Center (PA)
Grady Memorial Hospital (GA)
Guthrie Clinic Laboratories (PA)
Hahnemann University Hospital (PA)
Harris Methodist Fort Worth (TX)
Hartford Hospital (CT)
Health Network Lab (PA)
Health Partners Laboratories (VA)
Highlands Regional Medical Center (FL)
Hoag Memorial Hospital Presbyterian (CA)
Holy Cross Hospital (MD)
Holmes Regional Medical Center (FL)
Holzer Medical Center (OH) Hopital du Sacre-Coeur de Montreal (Montreal, Quebec, Canada)
Hôpital Maisonneuve – Rosemont (Montreal, Canada)
Hôpital Saint-LUC (Montreal, Quebec, Canada)
Hospital for Sick Children (Toronto, ON, Canada)
Hospital Sousa Martins (Portugal)
Hotel Dieu Hospital (Windsor, ON, Canada)
Huddinge University Hospital (Sweden)
Hurley Medical Center (MI)
Indiana University
Innova Fairfax Hospital (VA)
Institute of Medical and Veterinary Science (Australia)
International Health Management Associates, Inc. (IL)
Jackson Memorial Hospital (FL)
Jersey Shore Medical Center (NJ)
John F. Kennedy Medical Center (NJ)
John Peter Smith Hospital (TX)
Kadlec Medical Center (WA)
Kaiser Permanente Medical Care (CA)
Kaiser Permanente (MD)
Kantonsspital (Switzerland)
Kenora-Rainy River Regional Laboratory Program (Ontario, Canada)
Kimball Medical Center (NJ)
King's Daughter Medical Center (KY)
Klinični Center (Slovenia)
Laboratoire de Santé Publique du Quebec (Canada)
Laboratório Fleury S/C Ltda. (Brazil)
Laboratory Corporation of America (MO)
Laboratory Corporation of America (NJ)
LAC and USC Healthcare Network (CA)
Lakeland Regional Medical Center (FL)
Lancaster General Hospital (PA)
Langley Air Force Base (VA)
LeBonheur Children's Medical Center (TN)
L'Hotel-Dieu de Quebec (Canada)
Libero Instituto Univ. Campus BioMedico (Italy)
Louisiana State University Medical Center
Maccabi Medical Care and Health Fund (Israel)
Malcolm Grow USAF Medical Center (MD)
Martin Luther King/Drew Medical Center (CA)
Massachusetts General Hospital (Microbiology Laboratory)
MDS Metro Laboratory Services (Burnaby, BC, Canada)
Medical College of Virginia Hospital
Medicare/Medicaid Certification, State of North Carolina
Memorial Hospital at Gulfport (MS)
Memorial Medical Center (IL)
Memorial Medical Center (LA) Jefferson Davis Hwy
Memorial Medical Center (LA) Napoleon Avenue
Mercy Medical Center (IA)
Methodist Hospital (TX)
MetroHealth Medical Center (OH)
Michigan Department of Community Health
Mississippi Baptist Medical Center
Monte Tabor – Centro Italo - Brazileiro de Promocao (Brazil)
Montreal Children's Hospital (Canada)
Montreal General Hospital (Canada)
MRL Pharmaceutical Services, Inc. (VA)
Nassau County Medical Center (NY)
National Institutes of Health (MD)
Naval Hospital – Corpus Christi TX)
Nebraska Health System
New Britain General Hospital (CT)
New England Fertility Institute (CT)
New England Medical Center (MA)
New Mexico VA Health Care System
New York University Medical Center
North Carolina State Laboratory of Public Health
North Shore – Long Island Jewish Health System Laboratories (NY)
North Shore University Hospital (NY)
Northwestern Memorial Hospital (IL)
O.L. Vrouwziekenhuis (Belgium)
Ordre professionnel des technologists médicaux du Québec
Ospedali Riuniti (Italy)
The Ottawa Hospital (Ottawa, ON, Canada)
Our Lady of Lourdes Hospital (NJ)
Pathology and Cytology Laboratories, Inc. (KY)
Pathology Associates Medical Laboratories (WA)
The Permanente Medical Group (CA)
Piedmont Hospital (GA)
Pikeville Methodist Hospital (KY)
Pocono Medical Center (PA)
Presbyterian Hospital of Dallas (TX)
Providence Health Care
Queen Elizabeth Hospital (Prince Edward Island, Canada)
Quest Diagnostics Incorporated (CA)
Quintiles Laboratories, Ltd. (GA)
Regions Hospital
Reid Hospital & Health Care Services (IN)
Research Medical Center (MO)
Rex Healthcare (NC)
Rhode Island Department of Health Laboratories
Riyadh Armed Forces Hospital (Saudi Arabia)
Robert Wood Johnson University Hospital (NJ)
Royal Columbian Hospital (New Westminster, BC, Canada)
Saad Specialist Hospital (Saudi Arabia)
Sacred Heart Hospital (MD)
Saint Mary's Regional Medical Center (NV)
St. Alexius Medical Center (ND)
St. Anthony Hospital (CO)
St. Anthony's Hospital (FL)
St. Barnabas Medical Center (NJ)
St-Eustache Hospital (Quebec, Canada)
St. Francis Medical Ctr. (CA)
St. John Hospital and Medical Center (MI)
St. John Regional Hospital (St. John, NB, Canada)
St. John's Hospital & Health Center (CA)
St. Joseph Hospital (NE)
St. Joseph's Hospital – Marshfield Clinic (WI)
St. Joseph's Medical Center (NY)
St. Joseph Mercy Hospital (MI)
St. Jude Children's Research Hospital (TN)
St. Luke's Regional Medical Center (IA)
St. Mary of the Plains Hospital (TX)
St. Mary's Hospital & Medical Center (CO)
St. Vincent Medical Center (CA)
Ste. Justine Hospital (Montreal, PQ, Canada)
Salina Regional Health Center (KS)
San Francisco General Hospital (CA)
Santa Clara Valley Medical Center (CA)
Seoul Nat'l University Hospital (Korea)
South Bend Medical Foundation (IN)
Southwest Texas Methodist Hospital (TX)
South Western Area Pathology Service (Australia)
Southern Maine Medical Center
Spartanburg Regional Medical Center (SC)
Specialty Laboratories, Inc. (CA)
Stanford Hospital and Clinics (CA)
State of Washington Department of Health
Stony Brook University Hospital (NY)
Stormont-Vail Regional Medical Center (KS)
Sun Health-Boswell Hospital (AZ)
Swedish Medical Center – Providence Campus (WA)
Temple University Hospital (PA)
Tenet Odessa Regional Hospital (TX)
The Toledo Hospital (OH)
Touro Infirmary (LA)
Trident Regional Medical Center (SC)
Tripler Army Medical Center (HI)
Truman Medical Center (MO)
UCLA Medical Center (CA)
UCSF Medical Center (CA)
UNC Hospitals (NC)
University College Hospital (Galway, Ireland)
University Hospital (Gent) (Belgium)
University Hospitals of Cleveland (OH)
University of Alabama-Birmingham Hospital
University of Alberta Hospitals (Canada)
University of Colorado Health Science Center
University of Chicago Hospitals (IL)
University of Illinois Medical Center
University of the Ryukyus (Japan)
University of Virginia Medical Center
UZ-KUL Medical Center (Belgium)
VA (Denver) Medical Center (CO)
Virginia Department of Health
VA (Hines) Medical Center
VA (Kansas City) Medical Center (MO)
VA (Western NY) Healthcare System
VA (San Diego) Medical Center (CA)
VA (Tuskegee) Medical Center (AL)
Vejle Hospital (Denmark)
Washington Adventist Hospital (MD)
Washoe Medical Center Laboratory (NV)
Wellstar Health Systems (GA)
West Jefferson Medical Center (LA)
West Shore Medical Center (MI)
Wilford Hall Medical Center (TX)
William Beaumont Army Medical Center (TX)
William Beaumont Hospital (MI)
William Osler Health Centre (Brampton, ON, Canada)
Williamsburg Community Hospital (VA)
Winn Army Community Hospital (GA)
Winnipeg Regional Health Authority (Winnipeg, Canada)
Wishard Memorial Hospital (IN)
Yonsei University College of Medicine (Korea)
York Hospital (PA)

NCCLS ▾ 940 West Valley Road ▾ Suite 1400 ▾ Wayne, PA 19087 ▾ USA ▾ PHONE 610.688.0100
FAX 610.688.0700 ▾ E-MAIL: exoffice@nccls.org ▾ WEBSITE: www.nccls.org ▾ ISBN 1-56238-484-8

NCCLS