Validation of the Laboratory Information System

DANIEL F. COWAN, R. ZANE GRAY, BEVERLY C. CAMPBELL

The use of computer systems and computerized devices in the laboratory is becoming nearly universal, and increasingly processes once performed by people are being transferred to machines, the inner workings of which may be only dimly understood. The task of the people managing the work of the laboratory is to comprehend the processes by which the computerized laboratory information system (LIS) in use, usually acquired as a package from a vendor, may be shown to be doing the job it is expected to do. The functions of the system must be validated. Useful discussions of regulation¹ and inspection and accreditation² of laboratory information systems are available. However, despite its importance there is remarkably little published outside the blood bank literature about the validation of laboratory information systems that is accessible to general users of the systems, that is, addresses specifically laboratory issues in terms and concepts familiar to the laboratorian. Our intention here is to discuss LIS validation as it applies in the laboratory in general, to define the issues related to validation, and to suggest an approach to dealing with them.

Validation of the laboratory information system is the continuing process of proving the system fit for its intended use, initially and over time. Validation consists of defining, collecting, maintaining, and independently reviewing evidence that the system is consistently performing according to specifications. This is no small task, as many parts of a vendor-supplied LIS are proprietary, and some are so complex as to be beyond the ability of a user to evaluate in detail. Validation is resource intensive, and while certain aspects of validation testing are increasingly done with the support of automated utility programs, it remains a mainly manual operation involving many work hours. Difficult or not, this task must be undertaken, because it is an essential component of a quality management system, and the laboratory has a legal and ethical responsibility to assure itself and its clients that data

Modified from an article published in Arch Pathol Lab Med 1998; 122:239–244.
provided through the LIS are accurate, consistent, and reliable. Various professional accrediting associations require validation of the laboratory computer system, including the College of American Pathologists (CAP), the American Association of Blood Banks (AABB), and the Joint Commission on the Accreditation of Healthcare Organizations (JCAHO). The federal Department of Health and Human Services claims an interest, and the Food and Drug Administration (FDA) asserts jurisdiction under the Safe Medical Devices Act of 1990 over a computer system supporting the operations of a blood bank that produces blood products. The FDA in effect holds that such a blood bank is in fact a manufacturer of biologicals and is to be held to the same current good manufacturing practices (cGMP) as any such manufacturer. Computer systems must be validated to the FDA standard when they are used in conjunction with manufacturing, storage, or distribution of biologicals and drugs. Blood products are deemed to be biologicals. This introduces a new way of thinking into the laboratory, even for those used to meeting the highest standards of the laboratory peer group. Regulation concerns not only the clinical laboratory and the LIS more generally but also applies to in vitro medical devices, of which computerized cytology instruments are a prime example. Hence, regulation will be addressed along with the general issue of LIS validation.

**General Validation Issues**

**Validation for Quality Assurance and Improvement**

A quality management system addresses four questions: What is it that I do? How do I know that I do it well? How can I convince others that I do it well? How do I get better at what I do? While these are familiar in the context of the clinical and anatomical laboratories, they apply equally well to the LIS. What the information system does is collect, process, store, and distribute information. Validation is directed at the next three questions. It demonstrates to all concerned, in and outside the laboratory, that the LIS manages information well, with the expected accuracy and reliability, file integrity, auditability, and management control. Validation also identifies weak points or areas for improvement and, with proper planning and documentation, can show that the LIS has in fact gotten better or warn that adverse trends and conditions are developing in time to prevent them. It is important to recognize that system failure does not necessarily mean loss of operation; it can mean failure to provide expected services in a timely manner.

A laboratory information system is a specific application of a planned system of processes, operations, and procedures, part of which is supported by a computer system. A computer system includes not only the core computer, consisting of the hardware and its controlling software, but also people, peripheral equipment, documentation and defined procedures that relate to
3. Validation of the Information System

it. The central questions in validation are “Does this complex system work?” and ‘Do you control it?’ A computer-based system acts with nearly incomprehensible speed, handles enormous volumes of information, can broadcast it widely almost instantaneously, and must store it indefinitely in electronic form, not readable until it is displayed on a screen or as a printout. Validation of the system poses special problems that must be addressed by very carefully structured, highly documented processes.

It is important to recognize that computer programs, except for the most simple, no matter how carefully written and tested, are always defective to some degree. All software contains errors, or “bugs,” that may or may nor be evident in casual use but will inevitably show up with use of the program over time. It is also important to understand that programs do not deteriorate spontaneously or wear out. They stay the same until they are changed by some exogenous factor. However, under conditions of use, deeply buried flaws may surface, or a once functional program may not integrate well with newly added functions. The implication for validation is that a program is always suspect and must be tested in use under varying conditions and load over a long time, and the process and results of testing must be documented—an unending process.

In contrast, hardware may leave the manufacturer working perfectly as far as it can be ascertained, and the user may be satisfied by a routine of acceptance testing that it functions according to all specifications. Unlike software, some hardware deteriorates with use over time, and therefore a device that was at one time fully functional may become unreliable and even fail. The implication for validation is that hardware must be under a program of testing that will predict failure before it occurs, and in the event of a crash, data in process at the time will not be lost or corrupted and can be recovered with full confidence.

The process of validation naturally divides into developmental validation, acceptance validation, and retrospective validation. Developmental validation is integrated into the process of customizing, configuring, adapting, or inventing a new system or component. It is the responsibility of the vendor who offers the product for sale or license. Acceptance validation or acceptance testing is the initial approval process for live use of the system by the user. Retrospective validation is the determination of the reliability, accuracy, and completeness of a system in use. While this process is actually concurrent with use, the term “retrospective” is used by the FDA, we have adopted it to avoid confusion. Ideally, validation begins before the system is acquired by development of a plan and by systematic evaluation of the vendor and the vendor’s system of validation. The reality of the situation is, however, that as the process of validation becomes better understood, it is applied to systems in use. This raises an important question: If the system is working well and has given no problems, why do we have to go through the process of validation? One answer is that if you prepare blood products, the FDA says you have to validate. A better answer is that computer systems are
constantly being revised or adjusted and are therefore actually evolving systems, and not as static as they might seem. An appropriate process of validation prevents problems, and, if one does occur, mechanisms are in place to identify it and its cause, and to prevent a recurrence. Also, a proper process of validation identifies objectively when it is time to upgrade the system or its components.

**Vendor Validation Testing**

As part of the presentation of a product to a prospective client, a system vendor must document and attest that an appropriate program of validation has been carried out. No single agency regulates the development of computer-supported information systems, and there is no universal industry standard. Therefore, the user must have a reasonably clear idea of the validation processes used by a prospective vendor and must demand appropriate documentation. Although not required, some vendors have adopted the ISO 9000 standards of the International Organization on Standardization. The ISO is not a certifying body. If a vendor claims to follow ISO standards, the claim has to be certified by a third party.

The validation processes used by a vendor may be extremely detailed and organized at stages or hierarchical levels in the system. The same processes also apply to locally written systems. While the details are highly technical and beyond the scope of this discussion, they may be summarized briefly. The lowest level is proving that a single function within a program works as intended. This level is referred to as **string testing**, a string being a continuous set of alphanumeric characters that does not contain numbers used in calculations. An example of this level is the printing of a label. The next level up is testing of a functional module or unit (**unit testing**) of some complexity, such as accessioning of a specimen. Above that is the level of **module testing**, in which a product function as a whole is evaluated. Last is **system testing**, in which the performance of functions and modules working together as an integrated whole is proven. The methodologies used include testing the internal workings of code and individual statements and conditions within the program. As these are all known to the developer, this process is called “white box testing,” in contrast to “black box testing,” user-oriented testing of observable input and output from a process not known to the user. Specific test methods include equivalence class partitioning, boundary value analysis, and error guessing, used to validate the limits of a program or product, including expected and unexpected conditions, to assure that appropriate actions are taken. Regressive validation ensures that changes have no unexpected effect on existing functions.

It is difficult for a person not involved in the development of complex software products to comprehend the amount of specification documentation that lies behind the product. One vendor describes ring binders occupying
200 square feet, floor to ceiling, relating to blood bank products alone. The only really practicable approach to assess the developer’s validation practices is to include questions on these practices and any formal certification of them in a request for information (RFI) or request for proposal (RFP) and to include the vendor’s responses as a commitment in a final contract. Close examination of site-preparation manuals and technical manuals may give some insight, and, if finances permit, engaging an expert consultant to report on the prospective vendor’s practices may be valuable. An arm’s-length approach to the vendor’s source code may be a wise policy, as too close an inspection may imply in the minds of some regulators a duty to review it in detail, a task usually well beyond the ability of a laboratory staff.

**General Process of User Validation**

The general process of user validation includes several steps.

**Identification and Description of the System to be Validated**

To address identification and description issues, the intended functionality of the system must be specified and documented for reference. Identification of the system by name specifies and limits the scope of validation. The description of the system should include the primary functions of the system and the service needs it is to meet. It includes a statement about the functional environment of the system, that is, the systems that interface or communicate with it, and it specifies the users of the system. This includes other systems (e.g., a hospital information system) and instrument interfaces. It specifies the names of the component hardware and software vendors, the names of the persons responsible for operating and maintaining the hardware, the operating system, and the applications software, and the name of the custodian of the data. Hardware components are specified by name, model number, and vendor or supplier, and software components are specified by module name, version, or release and vendor. The description also includes software specifications; an estimation of the user, workstation, terminal device, memory, and storage requirements, and a description of each functional module; the interaction of the modules, and the interaction of the system to be validated with other systems. A graphical depiction of the system and its modules and interfaces is a useful summary document.

**Specification of the Stage in the System Life Cycle**

A computer system can be thought of as having a life cycle consisting of several stages. A stage in a system life cycle can be any one of a sequence of events in the
development process of an information system. It will be bought, used, upgraded, and perhaps eventually replaced. Validation tasks depend on the cycle stage of the system and include definition of requirements, system design specification, implementation, testing, checkout, and operation and maintenance. Specifying the stage in the system life cycle allows testing and validation to be conducted at distinct points of development of the information system.

**Formalization of the Process for Prioritizing and Deciding**

Validation of software is done according to priorities established in the laboratory reflecting perceived need. Setting of priorities may depend on whether the FDA has an interest in the laboratory. Ordinarily, the core operational components have highest priority, as their function is to capture, store, and make available common patient information such as demographics, order and specimen collection data, and charges. General laboratory functions also hold high priority but in most cases only for immediate patient care. If the laboratory operates a blood bank that processes blood products, the blood bank donor module is the primary concern. Blood bank modules also have the long data storage requirements, along with anatomic pathology and microbiology.

**Development of Hazard Analyses**

Hazard analysis is determination of the potential degree of human harm that might result from the failure of a device or system; i.e., specification of the penalty for failure. Failure may be as serious as issuing bad data or as minor—but annoying—as printing poor-quality paper reports. A hazard analysis is part of every procedure.

**Identification of Regulatory Concerns**

Pinpointing regulatory concerns will determine which regulatory agency, if any, will claim jurisdiction, and which set of standards will apply to the operation of the system and the validation process. The agencies may include the FDA, state health departments or licensing bodies, and professional accrediting organizations, such as the CAP, AABB, and the JCAHO.

**Documentation**

Validation of a system produces a large volume of documentation, including not only the general statements about validation and test plans but also test scripts and attached listings, screen prints, and test reports. A document must include an explicit statement of pass/fail status, the signature of the person conducting the test, and the person reviewing the test status and disposition.
The documentation must be compiled, organized, and kept in a designated location for management and regulatory review.

**Validation Plan**

The first step in user validation is the development of a validation plan that implements in a very specific way the general issues already specified. A validation plan defines the initial and continuing program to be followed by the laboratory and specifies the documents needed to support it. Some LIS vendors may participate actively in post-acceptance validation, and others may offer useful advice.

**Initial Validation Program**

*Compile all Documentation of Functional Needs and Specific Plans for the System*

Define the testing and operating environments and all system interrelationships, the size (capacity) of the system, and the data and samples to be processed. Include existing evidence of validation of components by the vendors of software and hardware. Specify the events and criteria for selective revalidation. Define significant hardware and software updates and a schedule for revalidation in the absence of change. Develop a plan to discover error and institute corrective action. Assign areas of responsibility to named individuals. Develop a cross-referenced filing system so that all interrelated documentation is available on short notice.

*Review the Vendor’s Record*

The vendor must present evidence that it has followed established standards for system development and maintenance. Attestation and evidence of vendor validation is an essential component of documentation. If a recommended standard (e.g., ISO 9000) has been used, then a third-party certification must be provided. The validated processes must be pertinent to the system under consideration. Reviewed vendor records, in addition to validation documents, may include licenses and agreements among component vendors, system documentation, system management, and operator and user manuals. Evidence of product support by the vendor, such as approach to problem reporting and resolution, the system used for assignment of priority to requests for assistance, and system for follow-through on reported problems should be presented. Review of vendor source code is seldom feasible, as it may be extremely voluminous, may require expertise not available to the buyer, or may be proprietary. The implications of too intimate a familiarity with the vendor’s source code have been touched on.
Review Vendor-Supplied Documentation for Software and Hardware

The vendor is expected to supply full documentation of the product supplied, including system specifications, technical manuals for the operators of the system, and applications manuals for the users, but not including the documents accumulated during the process of product development. Part of the validation process includes determining that all necessary manuals are available, useful, and on hand. As vendor-supplied documentation of the system as a whole may be excessive for the user in the laboratory, it may be supplied to the system manager, and operational summaries such as bench-level operator manuals may be provided or developed, which are reviewed for accuracy and completeness. Maintenance and error logs are kept. The existence and use of appropriate vendor-provided materials, operator manuals, and maintenance and error records is specified in standard operating policies and procedures (SOP), periodically reviewed and documented.

Review and documentation of source code is primarily a matter of concern to regulatory agencies, who hold the user responsible for the validity of the code. For reasons already given, code review is vicarious, and depends on certificates and attestation statements.

Decide Which Subsystems Must Be Validated and Prioritize the Systems to be Validated

While all systems should be validated to some degree, not every component must be validated to the style and standard of a regulatory agency. Some validation is done by the hardware vendor and the system vendor and reported to the user. It may be convenient to validate all components to the most rigorous standard to avoid confusion arising from parallel and unequal standards. Not every task can be done at once. Some are more critical than others, so rational priorities must be established, based on hazard analysis.

Review Laboratory-Developed Standard Operating Procedures and Maintenance and Error Logs

SOPs are procedural instructions, written in a standard form, that specify the steps required to perform tasks. The manual of SOPs includes the procedures for handling all operations, from normal events to recovery from catastrophic interruptions by any cause. It should include detailed descriptions of all validation protocols, routine operating policies and procedures, and disaster recovery procedures with detailed documentation of all test results, events, and actions taken and the results of tests and actions. Personnel qualifications and responsibilities are also included. SOPs are written to a level of detail that allows an appropriately trained person to execute the procedure as defined. All SOPs are initially authorized by a designated competent author-
3. Validation of the Information System

ity, written by a knowledgeable person, and reviewed regularly. Review and the results of review are documented. Since a large number of SOPs are created, it is recommended that a “rolling review” be instituted; for example, one twelfth of the procedures are reviewed each month to avoid an unwieldy and hasty annual review.

**Design Test Plans and Document Results**

Test plans are designed to prove that a system functions to the extent that its components work according to expectation and design. They identify requirements that assure that the systems perform as expected running in the installed environment. **Test scripts** identify the exact procedures for testing and documenting a successful or unsuccessful test. They are developed in conjunction with the test plan and must indicate the action taken when a test is not successful. Test plans and scripts should have corresponding numbering systems to facilitate cross-referencing. The elements of a test script include the **functional area** under test, the **specific program** or module under test, usually designated by a mnemonic, a **validation scenario**, which says exactly what is being done, **criteria for acceptance**, an **acceptance statement**, i.e., whether the tested element is accepted or not, and an **approval statement**.

**Develop a Security Plan**

A security plan defines the level of security for each user, typically by functional category, i.e., clerk, technician, physician. Security level and access are tightly controlled by the LIS administrator. Security level defines the functions that a user is allowed to perform and reflected in the security level assigned to the unique user identification number and password. Security is assigned according to a hierarchy, the most fundamental functions being READ and WRITE. READ means that a user can view but not change data. WRITE allows change of data. Further, a security level may allow a clerk to read and write demographics and read but not write test results. A pathologist may read and write, but other physicians may read only.

Apart from the issue of security and confidentiality of patient information is the security of the operating system and applications software and the hardware itself. Physical access to the computer and to stored tapes and other archival systems must be limited to appropriate people, who are identified in the SOPs. To keep unauthorized users out of the system software as well as away from patient information, policies and procedures are defined to prevent access to the software either inadvertently or by crackers gaining entry over network and modem lines.

As part of the vendor’s program of maintenance and support, it may be necessary for the vendor to log on to the system from a remote site over telephone lines. This capability must be controlled and documented as part
of the overall system of security. At one extreme, the vendor calls the system administrator for access each and every time access is needed. At another, the vendor has direct dial-in access, according to a defined plan. Each dial-in must be logged by time, date, and reason for the call; the person logging in, the action taken, and the results of the action recorded.

**Develop a Plan for Change Control**

Change control refers to keeping track of all revisions, alterations, and fixes applied to a hardware or software component. All changes made in a computerized system are formally authorized and documented. “Changes” include any replacement, alteration, or enhancement of the hardware or software or important operational routines. Procedures governing change should specify the need for change and identify the authorizer of the change, the various validation protocols to be followed, the process of change, including the module affected, the person making the change, the tasks needed to effect it, the date they are performed, the result of the change, and recognizing that change, however carefully planned, may have unanticipated effects, how much revalidation is necessary following the change.

**Provide and Maintain an Appropriate Safe Environment for the System**

Vendors of computer equipment generally define the environmental conditions necessary for optimal functioning of the equipment, including temperature and humidity and the amount of floor space necessary to allow access for maintenance and upgrade. Usually, warranties depend on the vendor’s prior evaluation of the proposed space. Periodic measurements of temperature and humidity are made and documented. The period is defined by the stability of the measured parameter over a length of time. Highly stable environments are measured less often than volatile ones. An alarm system warns of incidents. The alarm system most practically displays alert or alarm messages to an operator working at a computer console, who takes whatever action is specified by procedures. A hierarchy of warnings may be established, whereby a duty operator at a console is authorized to handle incidents of relatively low import, while more serious happenings are reported immediately to a manager. Alert buzzers or other noisemakers are usually neither needed nor appropriate if an operator is more or less continually at a console or frequently checks a display. They may have a use when a displayed notice is not acknowledged within a specified time period.

**Develop a Disaster Plan**

A disaster plan is a group of detailed instructions formulated as SOPs that define what is to be done in the event of a major malfunction or a catastrophic interruption of function. The first part of a disaster plan is a definition of a
disaster and specification of the kind of occurrence likely to befall the facility. The definition varies depending on the physical location of the computers and the history of the geographical region, including exposure to such events as blizzards, power outs, earthquakes, hurricanes, or tornadoes. The plan indicates what provisions are made for preservation of the record, including archives; how services will be supported during the period of downtime, how they will be brought back up, and who will perform each task. Any significant uncontrolled downtime must be followed by a period of selective revalidation to ascertain that all systems are working according to specification. Full documentation of the event and of all steps taken in the recovery period is mandatory. It is prudent to maintain a duplicate archive in a place not likely to be affected by the same catastrophic event. Currently, some laboratories are developing provisions for a “hot backup,” an alternative computing site to which operations are shifted in the event of catastrophic failure of the primary system. This may become a requirement for accreditation of a laboratory by the College of American Pathologists.

**Conduct Stress Testing**

In testing the system as a whole, it is important to know how the system performs within the extremes of its rated capacity. Commonly, a system is designed and implemented at a level of capacity well within the workload expected to be placed on it, and an expectation for growth is built in. It is not enough to show that a system is functioning reliably at the middle of its rated capacity, since sooner or later system work will approach the specified limits of capacity. A system designed to accommodate 150 concurrent users may be initially installed with only 100 users, but eventually, 150 users may have to be accommodated. Similarly, the number of tests or procedures put through a system may rise from a comfortable level to one that approaches or reaches the system’s design limits. It is expected that as capacity is reached, response times will be degraded; the system will slow down. It must be shown that the system functioning at its limits will not lose or degrade stored data. This is analogous to the idea of linearity in a clinical laboratory test. The process for evaluating this scenario is called *stress testing*. Since it is not feasible to load an operational system with data and users to stress-test it to the point of failure, various approaches are used. Some system vendors and after-market vendors offer testing programs to address this issue. Other strategies involve the use of vendor-supplied utilities delivered with a computer’s operating system, software scripting products, or simply putting enough extra people on line to approximate the rated capacity of the system. It is important to test loading situations with several or many staff members attempting to modify the same patient record at the same time to be sure that the record is not garbled or otherwise compromised.

Data is intended to be stored for a long time in the system. It must be shown that there is no loss of integrity of stored data over time. Integrity of the archive can be tested by creating one or more test patients for whom each and every test
offered by the laboratory is ordered and resulted. Thereafter, the records of these test patients are printed and compared with the reference record, i.e., the data as entered, to identify any loss or corruption of data. This test is run at designated intervals, and the results of the comparison are documented. A comparison study is done every time the system hardware or software is upgraded and at least quarterly. An upgrade includes introduction of new laboratory test procedures, installation of new software error correction routines or patches, and addition of new disk storage space or random access memory. Integrity testing is also done on recovery from an unscheduled downtime occurrence.

It is very important to demonstrate that after an unplanned, uncontrolled interruption in function (crash), the system, when brought back up, has neither lost or corrupted data as a result of the crash. A typical way of assessing the integrity of the archive is to print the test result records of the test patients and compare them with the reference records. Combined with tests of other more dynamic functions, such as entering orders, printing labels, entering results, and transmitting results, this procedure gives a quick sample of every function in the laboratory, including the ability to print records in the specified format.

**Continuing Validation Plan**

**Maintain an Inventory of Hardware and Software**

An inventory is a complete listing of all components of the system and the source and date of acquisition.

**Create and Maintain SOPs**

This task is familiar to managers of clinical laboratories. Knowledgeable individuals draft policies and procedures, which in effect define the information system. The SOPs are authorized, reviewed, and approved by competent authority in the laboratory. They are reviewed, changed, or continued as often as necessary, and at least once a year. Review is documented by signature, and the reasons for change, if any, are specified. Outmoded procedures are retired but maintained in an archive file.

**Test the Hardware and Software, and Document the Results**

All hardware and software are tested before acceptance and periodically thereafter. All such testing is according to plan as defined in the SOPs, and the results are documented. These tests, run under the general heading of system qualification, are done in three parts: on installation (installation qualification), at start, restart, recovery, and maintenance (operational qualification), and in performance of the procedures for which the system was acquired (performance qualification). A master test results file is maintained in a designated office. While parts of the
3. Validation of the Information System

documentation may be kept for convenience in sections of the laboratory, the master file governing all sections of the laboratory is kept in a single place.

**Maintain Current Hardware and Software Documentation**

Vendors provide elaborate volumes of documentation of both hardware and software. While duplicate volumes relating to particular functions may be kept in laboratory sections or a computer room, a complete master set is maintained in a designated office.

**Ensure That System Procedures Are Followed**

A comprehensive set of SOPs and vendor-provided user reference guides describe system procedures to be followed. It must be ascertained and documented that these procedures and guidelines are in fact followed. An automated operations cycle, which includes various tasks and activities to be completed on a scheduled basis, is used for the core functions of file and queue management, report and chart generation, and foreign system and instrument interfaces. The operations processes are most often performed in the background as computer-driven operations, ensuring that procedures are consistently the same. Procedures not carried out as part of operations cycles are documented and reviewed by supervisors at an appropriate level.

**Monitor Performance and Problem-Reporting Procedures**

Records of problems and steps taken to identify their cause and to fix it are kept. System administration assures that these records are kept.

**Run Backup and Restore Programs in Accordance with Procedures and Documentation**

Some of these programs are run as part of routine operations, according to a schedule. Others are run following unscheduled downtime. The performance of these procedures is documented.

**Maintain, Test, and Ensure Compliance with the Disaster Recovery Plan**

Disaster drills should be run periodically to ensure that all persons understand their responsibilities and that all systems function as they are supposed to. The results of these tests are documented.

**Train Personnel for Job Requirements, Including Familiarity with Policies and Procedures**

Training is crucial if a system is to be used appropriately and if all policies and procedures are to be followed. A formal training program, or at least a
program that orients new employees to the system and documents training in their areas of responsibility, is mandatory.

Record retention and organization is a very important issue, based on the principle that if you haven’t documented it, you haven’t done it. Validation records must be organized and accessible. This usually means cross-referencing test panels and test results, maintaining logs of maintenance and software changes, and the like. How long records must be kept is not all that clear. Some say two years, and we advise five years to meet the requirements of the most demanding agency.

Validation for Regulatory Requirements

Regulated systems are those that impact product safety and/or reliability and those that generate data or reports for submission to regulatory agencies. While we believe that our recommendations will validate a laboratory information system to a level that will attain the objectives of assuring that the system will consistently perform according to specification, we do not pretend to say that following our recommendations will automatically satisfy FDA standards for validation. That is for the FDA to determine. The FDA has provided some guidance in the matter.\textsuperscript{9,10} The FDA guideline definition says: “Process validation is establishing documented evidence providing a high degree of assurance that a specific process will consistently produce a product meeting its predetermined specification and quality attributes.”\textsuperscript{11} Although the vendor of a system should play a role in the complex and expensive task of product validation at the user’s site, because the FDA does not regulate every aspect of a vendor’s production, the agency assigns responsibility to the user to comply with cGMP relating to the production of biologicals. In practice, the user must ascertain and verify the vendor’s compliance status as part of the overall validation plan.

Chapter Glossary

acceptance validation (acceptance testing): Initial approval process for live use of a system by the user.

cchange control: Process of tracking all revisions, alterations, and fixes applied to a hardware or software component of a computerized system; all changes are formally authorized and documented.

developmental validation: Validation procedures integrated into the process of customizing, configuring, adapting, or inventing a new system or component; the responsibility of the vendor who offers the product for sale or license.

disaster plan: Group of detailed instructions formulated as SOPs that define what is to be done in the event of a major malfunction or catastrophic interruption of function.

hazard analysis: Determination of the potential degree of human harm that might result from the failure of a device or system, i.e., specification of the penalty for failure.

hot back-up: Alternative computing site to which operations are shifted in the event of catastrophic failure of the primary system.
**module validation** (module testing): Testing of the function of a product as a whole.

**process validation:** Establishing documented evidence providing a high degree of assurance that a specific process will consistently produce a product meeting its predetermined specification and quality attributes.

**regressive validation:** Testing process that ensures that changes in a system have had no unexpected effect on existing functions.

**retrospective validation:** Determination of the reliability, accuracy, and completeness of a system in use (same as concurrent validation).

**SOPs** (standard operating procedures): Procedural instructions, written in a standard form, that specify the steps required to perform tasks.

**stress test:** Form of testing in which a load approximating the maximum designed work capacity is placed on the system to determine whether a system operates with expected reliability at the limits of its capacity.

**string:** Continuous set of alphanumeric characters that does not contain numbers used in calculations.

**string testing:** Proving that a single function within a computer system works as intended.

**system validation** (system testing): Testing of the performance of functions and modules working together as an integrated whole.

**test script:** Description of the exact procedures followed for testing and documenting a successful or unsuccessful test. Scripts are developed in conjunction with the test plan and must indicate the action taken when a test is not successful.

**unit validation** (unit testing): Testing of a functional module or unit of some complexity.

**validation:** Continuing process of proving an information system fit for its intended use, initially and over time.

**References**

7. International Organization for Standardization. 1, Rue de Varembé casse postale 56 CH-1211, Geneva 20, Switzerland