Validation of Spectrometry Software

Part I

R.D. McDowall

This column, the first in a series dealing with validation of spectrometry software, provides an overview of the concepts of validation, business versus regulatory rationale for validating software, and some of the common principles and wrong assumptions of computer validation. The system development life cycle is defined and some of the key documented evidence required for validation is explained.

You might wonder why am I writing a series of columns about software validation for spectrometry instruments. The fact that I'm getting paid for them is a rather poor reason from your perspective but okay from mine. However, the real reason is to educate you and give you a better perspective of the business and regulatory reasons for software validation under the current regulatory climate, as well as an overview of the system development life cycle and documented evidence.

Because many different types of spectrometers are used within regulated industries, I've concentrated on the software elements rather than the operation of the instruments. That is not to negate the role of the instrument itself; the purpose of this series is to look at the process of software validation that should accompany instrument qualification. Many of you will be familiar with qualification of the instrument but not with software validation, thus the emphasis on the software elements.

In this series, I want to discuss the prospective validation of spectrometer software. By prospective validation, I mean undertaking the validation work in parallel with the life cycle of the project as it progresses from start to finish. Unfortunately, this is not always possible. Usually just before the system goes live, someone thinks that perhaps the system should be validated. Taking this approach will add 25–50% to the validation costs of the project, mainly because documentation that should have been written at key stages of the project was missed or, if written, was not of sufficient quality for laboratories working under regulations such as Good Manufacturing Practice or Good Laboratory Practice.

WHY BOTHER TO VALIDATE YOUR SOFTWARE?

Let's start at the beginning and ask this fundamental question, as there are a number of reasons for validating your spectrometry software.

Investment protection. How much money does your laboratory waste buying software that is not up to scratch? The investment in spectrometers operating in the laboratory, like those used for raw materials testing in the warehouse, has risen dramatically during the past decade. How successful have these purchases been? Validation is a way of building quality into a system; it increases the odds that the spectrometer and its software will meet expectations. Therefore, the investment that an organization makes is protected from purchase on a whim or, worse, from the end-of-year slush fund spend. You know the sort of thing — your boss puts his or her head around the door and asks if you can spend $100,000 in three weeks (get three competitive quotes, assess the systems, raise the purchase order, and have the empty box delivered to stores by the end of the financial year). Or does this always happen in other organizations?

Consistent product quality. The term *product quality* can be used in the widest scope: The product of a laboratory is information to make decisions. Therefore,

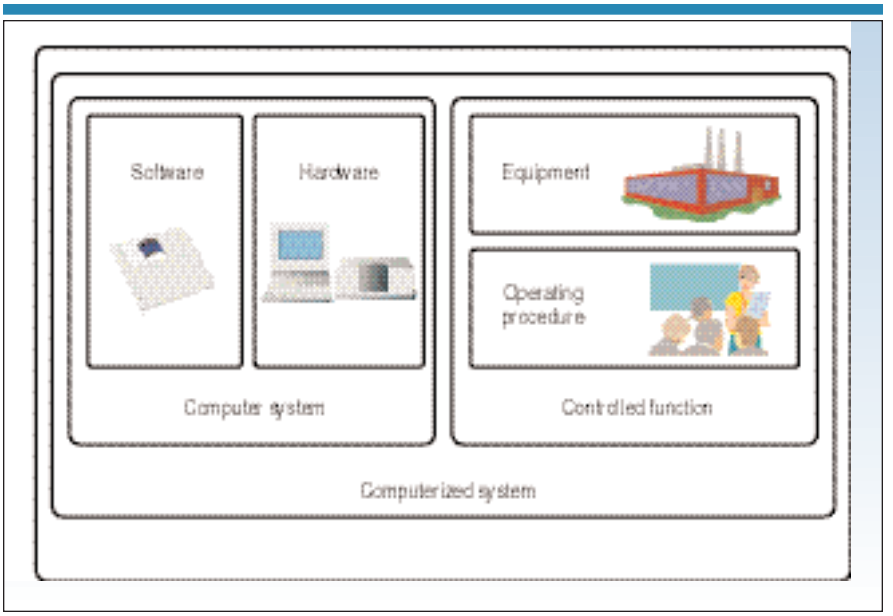

Figure 1. Elements of a computerized system.

from R&D laboratories, software validation is used to ensure that the results you generate to support product development are correct. A spectrometer can also be heavily involved with manufacturing, and it is important to know that data used to release a product or accept raw materials is also correct and can help to ensure consistent quality of the final product.

Compliance with regulations. Both the FDA (1) and the European Union (2) expect manual and computerized systems to show equivalent quality. Good validation practices will ease or expedite regulatory inspections and audits and reduce the risk of noncompliance. Confidence in computerized data enables a good foundation for management control, especially throughout a multinational company, that can be evidenced with better communication across teams and with regulators. Furthermore, emphasis on the electronic records and electronic signatures final rule that we discussed in previous installments in this series (3–5) affects spectrometers. You'll notice from surfing the FDA web site that compliance with these regulations is voluntary; however, with most spectrometers having software, you have already volunteered for compliance with the electronic records part of these regulations.

KEY TERMS AND CONCEPTS OF COMPUTER VALIDATION

We need to get a number of terms and concepts right before we start the detailed journey into validation of your software in the following installments in this series.

What is validation? Process validation is defined as *establishing documented evidence which provides a high degree of assurance that a specific process will consistently produce a product meeting its predetermined specification and quality attributes* (6).

This definition was modified by the PDA for a computerized system: *establishing documented evidence which provides a high degree of assurance that a specific computer-related system will consistently produce a product meeting its predetermined specifications* (7).

The key concepts in the last definition above are

- documented evidence
- high degree of assurance
- predetermined specification.

There are other regulatory or quality guidelines from the European Union (2) and the Organization for Economic Development (8). Each regulation may have slightly different requirements, but all come down to the same series of requirements: In general, validation is concerned with generating the evidence to demonstrate that the system is fit for the purpose for which you use it, it continues to be so when it is operational, and there is sufficient evidence of management control. This usually means that an action must be documented. Another feature of validation is to produce an auditable system — having the appropriate documentation to aid any audit or inspection.

The problem is how to respond to the requirement for computer validation. Any response should

- be scientifically sound
- be structured
- provide adequate compliance
- reflect the way you use the application.

This latter point is most important — *there is no point in validating a function of a system that is not used.* Equally important is the fact that one laboratory's use of spectrometer software can be markedly different from another laboratory's use of the same software.

Computer validation must give laboratory managers and users confidence in the system first and foremost, second to an internal quality audit, and third to an external inspector. Inspectors audit the laboratory on a periodic basis; all others work in the laboratory and use its computerized systems daily. The users must have the confidence in a system above all others; otherwise, your investment will be wasted.

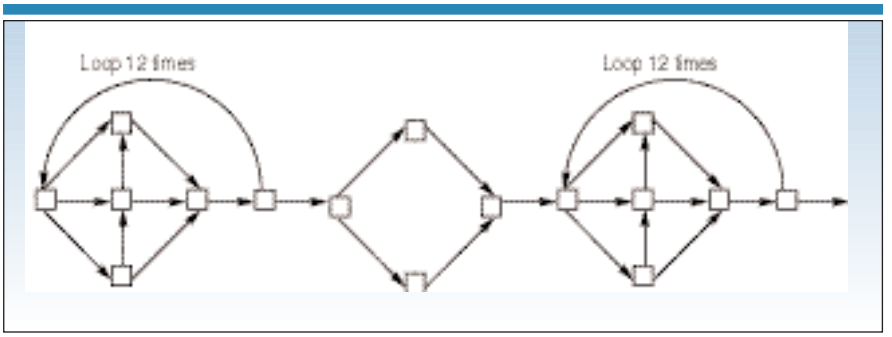

Figure 2. Complete testing of software is impossible (10).

WHAT IS A COMPUTERIZED SYSTEM?

Virtually all spectrometers used in the laboratory or in a production environment are classified as computerized systems. Figure 1 shows the key components (7). It is important to realize early in your project that if you are validating a computerized system, you don't just concentrate on the computer hardware and software; validation encompasses more.

The elements that make up a computerized system are a computer system and controlled function within the context of its operating environment. The computer system consists of the following elements.

Hardware. The elements that make up this part of a computerized system are the computer platform that the spectrometry software runs on such as workstation or server plus clients, and so forth; and any network components such as hubs, routers, cables, switches, and bridges. The system may run on a specific segment of a network or over a general segment of it, along with any peripheral devices such as printers, plotters, and connecting cables.

Software comprises several elements, such as

- operating systems of the clients and server
- network operating system
- application software (spectrometer software) and any utility software such as a database or reporting language.

The controlled function comprises the following.

Equipment linked to the computerized system, such as the spectrometer itself. The interface can vary from a simple transmission of absorption to more complex spectra acquisition. Ideally, the equipment connected to the data system should be qualified as part of the overall validation of the software; otherwise, how would you know that you are generating quality results?

Written procedures. Trained staff should follow written standard operating procedures (SOPs), as well as manuals, to op-

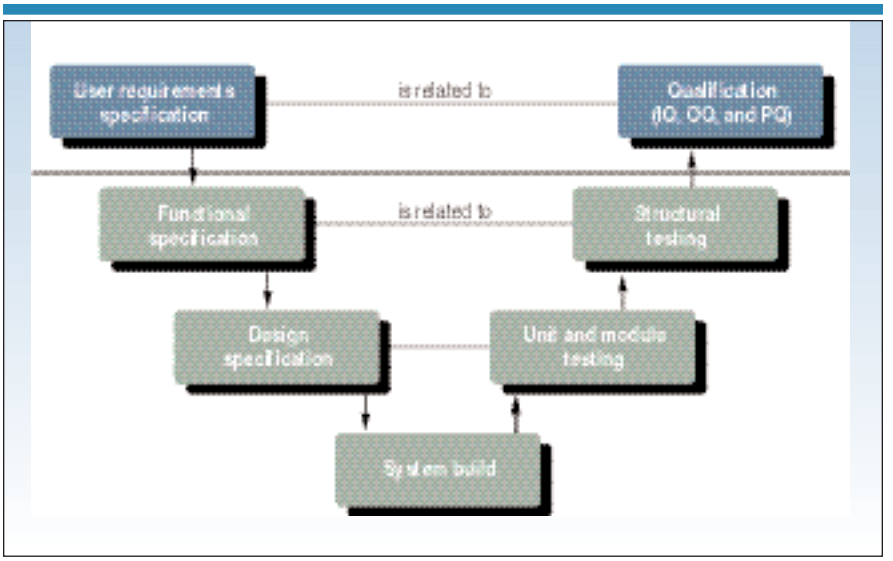

Figure 3. System development life cycle.

erate the equipment and the data system software.

To repeat, you must realize that validation is not just a matter of testing software, calibrating, or testing the spectrometer; a greater range of items needs to be considered under the scope of validation. To qualify your instrument and not validate the software leaves you open to regulatory action.

PRINCIPLES OF COMPUTER VALIDATION

A number of principles should be followed correctly during validation. A summary of the main ones follows; these are intended as practical issues that have arisen in the validation of computerized systems.

System owner is responsible for validation. The business owner of each system is responsible for the validation of that system. Although others may carry out validation on behalf of the system owner, the responsibility for validation cannot be delegated.

Risk assessment. A key consideration at the start of any computerized system project is, "Does the system have to be validated?" If the system is to be used to generate regulatory data, then validation is required; however, if it is used for research purposes only, then validation is not required. Undertake a formal risk analysis and document the result.

Team approach. Validation generally requires support from various functions and levels within the organization — for example, scientists involved in using a system, system owner, quality assurance and, if the system is networked, the information technology (IT) department staff responsible for maintaining the server. All roles involved in validation must take responsibility for validation.

Validation plan. There must be a formal and approved validation plan for each system. This plan needs to be written as early in the project as possible to avoid additional validation costs involved by writing documentation retrospectively that should have been written at the time the activity occurred.

Document activities. All activities must be documented in reviewable forms. Today this documentation can be in either paper or electronic formats. It is not enough to observe the result of an activity or test. The politically correct term for this approach is "informally documented"; this leads to regulatory observations and warning letters.

The four eyes principle. All documents should be written and reviewed by at least two people (or two sets of eyes) to ensure that they are correct from both the technical and compliance perspectives.

Document your requirements. Your user or system requirements specification is your map through the system development life cycle. It prevents you from being seduced by technology or salespersons. Without this document, you cannot validate a computerized system.

Traceable requirements. All functions and components of a system must be trace-

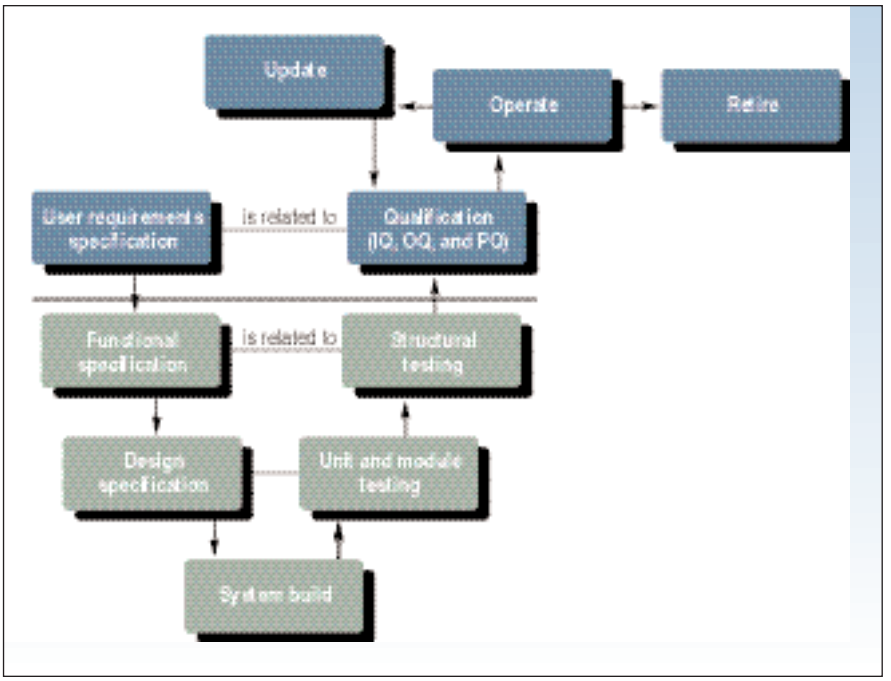

Figure 4. Modified system development life cycle to include operation, maintenance, and retirement.

able to approved specification document(s). Furthermore, it must be demonstrated that these requirements are met within the implemented system.

Vendor assessment/audit to assess software quality. Vendors must be assessed and, if necessary, audited. It is not adequate that another organization has audited the vendor; this must be performed by your organization. Furthermore, it cannot be assumed that products purchased from vendors are validated; all products (including hardware, software, and services) purchased must be checked for validation according to approved procedures.

Predefined test results and acceptance criteria. All testing must be based on comparison of actual results to expected results within defined and approved test scripts. Furthermore, acceptance criteria must be explicitly stated, not implied, and based on sound scientific principles.

Documented operation. The documentation must show that the operation of a system follows the system SOPs, identify which SOPs must be followed by the users, and reflect current working practices with the system.

Independent approval. The person approving key validation documentation must be independent of the validation team, the users, and the developers of the system. A quality assurance involvement from the beginning of the project is essential.

Organized archive. An archive for validation documentation must exist and it must be well organized. It must be possible for users to retrieve both physically and electronically archived documents accurately and quickly, or within 24 h as a worst-case scenario. This is essential to meet the requirements of *21 CFR 11* regulations.

Training and ongoing training. It must be demonstrated that all users, managers, technical support people, and IT operations staff are trained in and are familiar with the system, as well as applicable regulations, on an ongoing basis. This will require initial and ongoing training for all types of users (system manager, supervisor, user, and IT support staff).

Standard operating procedures. The system must be operated using documented and approved SOPs. Further, and crucially, it must be possible to demonstrate that users continue to use the documented SOPs over time.

Change management. Formal change management and configuration management procedures must be applied to all configuration items of the system, such as hardware, application software, system software, training materials, SOPs, and all documentation.

System access defined. Logical and physical access to the system, functions, and data must be clearly defined and validated. This needs to be updated regularly for compliance with *21 CFR 11* regulations if changes are made to be accessible by users or if functionality is increased.

Maintain validation. Once validated, a system does not automatically stay that way. The system owner needs to ensure that the system remains under control. Changes need to be validated or revalidated when they occur, after the impact of the change has been assessed. Moreover, the system may need to be audited internally to ensure that the validation status has not changed.

ASSUMPTIONS AND MISCONCEPTIONS OF VALIDATION

Many people are familiar with validation in general terms; therefore, a range of assumptions exists about it — many of which are incorrect or false. To help avoid these misconceptions, I will address some of the more frequent ones in this section.

We bought a validated system. False! Any vendor product implemented in a particular environment becomes a unique item because the combination of environment, parameters, configuration, data content, interfaces, user procedures, and so forth are unique. Remember that the system owner is responsible for validation and that this cannot be delegated.

The use of certificates of "validation" from vendors or "validation kits" apply only to the portion of the system development life cycle for which the vendor is responsible. The system owner is responsible for the whole life cycle and, at best, these materials provide a partial solution.

Partial validation of the system. You cannot partially validate a computerized system — it is either all (validated) or nothing (unvalidated). See the FDA draft guidance document of General Principles of Software Validation for further information (9).

Long-term use equals validation. The fact that a system performs without problems for an extended length of time does not mean that the system is validated. To be validated, a system requires documented proof that it meets predetermined validation criteria. See also comment 65 of the *21 CFR 11* preamble (10) for further details.

Validation is a one-off activity. Validation is not a single event in a system's life cycle because changes to the system inevitably occur — for example, upgrade of the application software or operating system. Therefore, ongoing revalidation of a system is required until the system ceases operation.

The data generated by the system need to be available for a batch's shelf-life plus one year for GMP data or 15 years after the last launch in the last country for a system operating in research and development, especially in the light of *21 CFR 11*.

Validation does not need documentation. Oh, yes it does! All activities contributing to validation of a system must be proven to have taken place — that is, documented either in paper or electronic

means. If it's not written (approved and reviewed), it's a rumor.

GMP = *g*iant *m*ass of *p*aper. The documentation needed to validate a system is little, if anything, more than that required for good practice delivery of a computerized system not requiring validation. Furthermore, references to vendor documentation can be used when these references include author, title, date/release number, and location.

Validation equals software testing. Wrong again! First, a system includes more components than just software — for example, procedures, hardware, documentation, and people. Second, activities other than testing are needed to prove system functions as desired — for example, system specifications.

Requirements are not needed. The definitions of validation above explicitly state that system requirements are required. In the absence of requirements:

- We cannot be certain which functions to specify to meet business needs.
- A system cannot be qualified to see if it meets these business needs.

Just a documentation exercise. It is not adequate to simply document validation features retrospectively. Validation must be proactively specified into a system. Furthermore, it must be demonstrated that using the system in practice continues to meet designed-in validation features — for example, that SOPs are being followed.

Validation is a job for IT or quality assurance/quality control. Validation is the responsibility of the users of the system — in particular, the system owner who is legally responsible for validation. You can't delegate this responsibility except in the case of incompetence.

Regulatory bodies don't care about IT systems. Wrong yet again! There are increasing trends both to inspect IT systems and in the level of sophistication of such inspections. This situation is especially true with *21 CFR 11* regulations and the resulting warning letters.

COMPUTER VALIDATION ROLES AND RESPONSIBILITIES

There are three key roles in validating a spectrometer data system from a laboratory perspective: the users, quality assurance, and IT, where appropriate. I will describe each role, along with an outline of its responsibilities.

Users: responsible for the overall validation of the system, which is achieved by defining the system's functions, selecting the system, verifying its installation, and defining and executing the validation plan. Users will need to have SOPs written for operating and supporting the application, the user base must be trained, and users must ensure that the complete documentation of the system is available for audit and inspection. Although end users are responsible for these areas, they need help, advice, and support in this. Active support by management is essential for making the resources available for the validation effort and to take the responsibility for authorizing the use of the system in the regulated environment. Furthermore, management personnel must encourage the participation of the quality assurance (QA) staff in this process.

Quality assurance: responsible for assistance in interpreting regulatory guidelines for computerized systems and how they apply to the spectrometer software. QA will review the key documentation produced during the validation effort. Monitoring of the testing and validation efforts and offering assistance in developing SOPs are additional roles and responsibilities for QA staff. If there are any vendor audits to be undertaken, then QA personnel should be involved in the planning and execution of this activity. Some QA personnel may not be very computer literate, but this must change as many regulations involving computerized systems require the active involvement of the QA staff.

Information technology: responsible for help in the purchase, installation, and operation of the spectrometer software for systems running on a network. If a group is not available or the users take on this role, then the responsibilities outlined here will be transferred to the users. Responsibilities will include running the hardware and software, backups, resolving problems, and so forth. However, in offering support for a regulated spectrometer software, the IT group becomes bound by the regulations or guidelines under which the laboratory works. What is not often realized both by the users and the information services group is that any unauthorized change to the operating system or network will make a validated spectrometer software noncompliant. We'll come back to this area in the next installment in this series.

External roles may also be involved, including:

• System vendor: The system vendor should help with advice on the sizing of the system, hardware needed for good performance, assistance with vendor audits, and help with qualification of the system (installation qualification and operational qualification only).
• Consultants for advice on the overall validation process or specific portions of it.

PROBLEMS WITH VALIDATION

A number of problems with validation exist.

Self regulation. Regulatory agencies take the view that the end users of a spectrometer software are responsible for its validation. The agencies will audit the system and will inform you if there are any problems with the work you have done. This is not satisfactory because the end users can rarely perform more than black box testing unless they have detailed knowledge of the design specification of the system and the aid of skilled computer scientists.

What am I to do? This leads to the problem of how to interpret the guidelines in a cost-effective approach to validation. Often many iterations of trial and error can be involved, wherein validation is either overengineered or not sufficiently rigorous.

Complete testing of a system is a myth. Unless the system is very simple, it cannot be completely tested. This was demonstrated by the work of Boehm (11) who described the simple program flow segment shown in Figure 2. The number of conditional pathways, and therefore possible tests, of the software in this segment was calculated to be 10^{21}. If one makes an absurd assumption that one test per second can be conceived, designed, executed, and documented, then it will take more than three times the geological age of the earth to validate this program segment. Unfortunately, most spectrometer software is far more complex, therefore procedures to record and fix errors are very important, as we'll discuss in a later installment of this series.

Consistency of audit. The human element, in the form of what will pass without comment with one inspector or auditor but not another will never completely disappear. The computer literacy of inspectors is increasing and with this will come increased scrutiny of computerized systems, including spectrometer software, far more so than now. However, consistency of regulatory approach and inspection is highly desirable.

FDA FORM 483 AND WARNING LETTERS

To gain a greater understanding of regulatory requirements, either a quick Internet trip to www.fda.gov or reading of selected issues of the *Gold Sheet* (12) is highly recommended. In the electronic reading room is a list of warning letters issued since 1996–97. Here you can see a Regulatory Authority in action. The citations associated with computer validation can be grouped into six categories (13):
• evidence of management responsibility
• evidence of system design and control of the design
• evidence of testing
• evidence of training
• evidence of audit and review
• evidence of document control.

Validation must address all of these issues, not only during the development of a system, but also during its operational life.

SYSTEM DEVELOPMENT LIFE CYCLE AND VALIDATION

Several models of the system development life cycle (SDLC) exist. The best one in my opinion is shown in Figure 3 and is known as the V model. A number of features are implicit in this, as explained here.

Design, build, and test. The first feature is the very simple concept of designing, building, and testing the application. The left-hand side of the V is concerned with designing the application; in our case the spectrometer software, but the model applies to any application. The bottom of the V is the system build: the programming of the units and modules and the right-hand side covers the stages in the testing and user acceptance (qualification) of the application.

Individual stages. Figure 3 also shows the individual stages of the V model life cycle:
• User requirements specification: specifies what the user wants the system to do. This is the basis of the user acceptance testing and qualification of the system.
• Functional design: This takes the user requirements and turns them into a computer programmer's view of the design for the system. This important stage requires the crossing of disciplinary boundaries: scientist to computer programmer.

From a vendor's perspective, the requirements of many laboratories are taken and incorporated into this document so that they can produce a product having as much appeal as possible to a wide range of potential customers.

- Design specification: further decomposition of the system design into individual units and modules of code. The function of each will be described, the inputs and outputs defined, and the integration of all to produce the overall system.
- System build: the actual programming of the system, which should involve programming standards to ensure that the code can be easily maintained and updated in the future.
- Unit and module testing: As each unit and module is completed it should be tested, first by the programmer who wrote it and then by a second independent person. As units are integrated into modules, the modules will be tested and some of the unit rests reapplied to see if functions have changes (regression testing).
- System testing: When all modules have been assembled into a system, a build version is usually tested in-house (alpha testing). When the vendor is reasonably happy with the functions, it will be released to selected users for beta testing and feedback. When all functions are working and it is relatively bug-free, the build is formally released as the next version and is available for distribution.
- Qualification: The new software is installed and the users test or qualify it to see if it is fit for purpose.

Time spent per stage. The model in its simplest form illustrates a flow down the left-hand side and up the right-hand side of the model. No feedback loops are shown in any of the figures illustrating the model in this article; however, do not be misled — feedback loops exist. The number and extent of them depend on the time the user or vendor has spent on the various stages. The more time spent in the design stages means that the build and test will go more smoothly and quickly.

Rushing the design to meet a deadline may mean that items are incorrectly specified or left out, which may not be discovered until the test stages of the application development. Design or specification of a spectrometer is the inexpensive part of the life cycle for both the chemist and the vendor. Missing or cutting short this stage means that either or both parties pick up the bill!

Relationships between stages. The V model is very useful for highlighting the relationships between the stages of the life cycle. Figure 3 shows these in outline: at the horizontal level there is a relationship between the design side to the testing side. For example, the design phase is related to the corresponding unit and module test phase. The design specification will outline the individual units and modules that will be coded, along with their functions. It will also outline which individual units and modules will combine (inputs and outputs between them, and so forth) to form the whole system. Therefore, the unit and module tests that are applied will base their test design and acceptance criteria on the specifications in the design document.

This relationship also applies to the other horizontal pairs in the model: functional specification and system test, and user requirements specification and qualification. This last pair is important from the perspective of the users because the User Requirements Specification defines the tests and their limits to be carried out in the qualification or user-acceptance testing. We will discuss this later in the series.

The design stages on the left-hand side of the model represent a decomposition of the problem: requirements in the URS are broken down into functional requirements and then into design requirements for individual units and modules of defined function. After programming, the test stages of the model on the right-hand side illustrate the building from these modules into a system ready for user-acceptance testing.

User and supplier responsibilities: It is unlikely that you will be developing your own spectrometer software; therefore, you will be purchasing a commercial system from a vendor. The V model is very useful in highlighting the responsibilities between the two parties. The users are responsible for the URS and the qualification or user-acceptance tests, while the vendor is responsible for the remaining stages of the life cycle. This is illustrated in Figure 3 by the horizontal line: above the line is the user's responsibility and below it is the vendor's. Again, we'll return to this area in later columns.

Limitations of the V Model. Although the V model looks good, there are some limita-

tions to its use. As you can see, the model only covers the initial development of a system from the user specification until it becomes operational. For this it is very good, but it represents only a small fraction of the total time that a system is used. Some data systems can be operational for 10 or more years (including upgrades of the spectrometer software and the hardware platform); therefore, there should be a mechanism to accommodate this in the model.

The following life cycle phases are missing from the V model:

- Operation: A number of tasks such as backup, recovery, change control, configuration management, archiving, and restoration need to be covered in this part of the life cycle.
- Maintenance: Every time an upgrade or change to the system is considered, this part of the life cycle will be invoked.
- Retirement: When a system is finally retired and the data are moved to a new system or archived, there is no mechanism in the current model.

Therefore, to accommodate these later phases of the life cycle, the V model could be modified to look like Figure 4.

For the documented evidence we need for validation, we can take this SDLC model and map onto it the documentation that could be produced in the life cycle and show their relationship to the life cycle. The documents that could be produced are listed below. The key ones are discussed in more detail in the next installment of this series; Table I provides an outline description of the function of each document.

- validation plan
- project plan

Table I. Validation package documentation.

Document name	Outline function in validation
Validation plan	Documents the intent of the validation effort throughout the whole life cycle Defines documentation for validation package Defines roles and responsibilities of parties involved
Project plan	Outlines all tasks in the project Allocates responsibilities for tasks to individuals or functional units Several versions as progress is updated
User requirements specification (URS)	Defines the functions that the spectrometer and software will undertake Defines the scope, boundary, and interfaces of the system Defines the scope of tests for system evaluation and qualification
System selection report	Outlines the systems evaluated either on paper or in-house Summarizes experience of evaluation testing Outlines criteria for selecting chosen system
Vendor audit report and vendor quality certificates	Defines the quality of the software from vendor's perspective (certificates) Confirms that quality procedures matches practice (audit report) Confirms overall quality of the system before purchase
Purchase order	From vendor quotation selects software and peripherals to be ordered Delivery note used to confirm actual delivery against purchase order Defines the initial configuration items of the spectrometer software
Installation qualification (IQ)	Installation of the components of the system by the vendor Testing of individual components Documentation of the work carried out
Operational qualification (OQ)	Testing of the installed system Use of a vendor's protocol or test scripts Documentation of the work carried out
Performance qualification (PQ) test plan	Defines user testing on the system against the URS functions Highlights features to test and those not to test Outlines the assumptions, exclusions, and limitations of approach
PQ test scripts	Test script written to cover key functions defined in test plan Scripts used to collect evidence and observations as testing is carried out Documents the predefined acceptance criteria and if they have been met or not
Written procedures	Procedures defined for users and system administrators Procedures written for IT-related functions Practice must match the procedure
User training material	Initial material used to train super users and all users available Refresher or advanced training documented Training records updated accordingly
Validation summary report	Summarizes the whole life cycle of the system and software Discusses any deviations from validation plan and quality issues found Management authorization to use the system

- user requirements specification
- system selection report
- vendor audit report
- vendor quality statement
- purchase order
- installation qualification documentation
- operational qualification documentation
- performance qualification test plan
- performance qualification test scripts
- SOPs
- user training
- validation summary report.

Taken together, this documentation will provide the validation package to support the contention that the spectrometer and its software are fit for purpose. Note that this is a suggested minimum list; you may write fewer or more documents than outlined here. The extent that you differ will depend on the amount of regulatory risk that the organization or laboratory management wishes to carry.

REFERENCES

(1) Good Manufacturing Practice regulations, in 21 CFR 11, "Electronic Records; Electronic Signatures, Final Rule," *Federal Register* 62 (1997) 13430–13466, World Wide Web: www.fda.gov.

(2) *Good Manufacturing Practice for Medicinal Products in the European Community, Annex 11* (Commission of the European Communities, Brussels, Belgium, 1997).

(3) R.D. McDowall, *Spectroscopy* **15**(5), 30–37 (2000).

(4) R.D. McDowall, *Spectroscopy* **15**(10), 26–31 (2000).

(5) R.D. McDowall, *Spectroscopy* **15**(11), 24–29 (2000).

(6) *Guideline for Process Validation* (Food and Drug Administration, Washington, D.C., 1987).

(7) *Validation of Computer-Related Systems,* Parenteral Drug Association Technical Report 18, 1995.

(8) *Application of GLP Principles to Computerised Systems* (Organization for Economic Cooperation and Development, Paris, France, 1995).

(9) *General Principles of Validation* (Food and Drug Administration, Center for Drug Evaluation and Research, Rockville, MD, 1997).

(10) 21 CFR 11, "Electronic Records; Electronic Signatures, Final Rule," *Federal Register* 62 (1997) 13430–13466, World Wide Web: www.fda.gov.

(11) B.Boehm, *Some Information Processing Implications of Air Force Missions 1970–1980* (The Rand Corporation, Santa Monica, CA, 1970).

(12) *The Gold Sheet* **34**(10) (FD&C Reports, Washington, D.C., 2000).

(13) S.Weinberg, "LIMS Validation," in *Laboratory Information Management Systems: Development and Implementation for a Quality Assurance Laboratory*, M.D. Hinton, Ed. (Dekker, New York, 1994).

R.D. McDowall *is visiting senior lecturer in the Department of Chemistry at the University of Surrey, principal of McDowall Consulting (Bromley, United Kingdom), and "Questions of Quality" column editor for* LCGC Europe, Spectroscopy*'s sister magazine. Correspondence may be addressed to him at 73 Murray Avenue, Bromley, Kent BR1 3DJ, UK.* ◆