Paper 154-2007

The Art of Code Validation

Mindy S. Rodgers, Eli Lilly and Company, Indianapolis, IN
Gregory C. Steffens, Eli Lilly and Company, Indianapolis, IN

Abstract

Art is a negotiation between the artist's imagination, talent, and medium. The art of code validation requires objective program assessment and willingness to make changes. Validation is of paramount importance in the pharmaceutical, biotech, and medical devices industries. We create what we think is our best work, frame it, and place it in the gallery to be shown—but does it fulfill regulatory obligations and comply with internal quality requirements? The degree of validation performed must be based on code complexity, consumer base, and carefully assessed risk factors.

15th century Europe considered their greatest artists merely craftsmen—great works were considered *collaborative* efforts. Shouldn't we approach code validation the same way? Our palettes must contain intersecting skillsets encompassing quality insight, IT expertise, development lifecycle methodology knowledge, and business intelligence. We should acknowledge areas in which we are deficient and solicit help from others to execute the process effectively (even Michelangelo had assistants while painting the Sistine Chapel).

Standardized validation methods increase efficiency and protect the time needed to invent. Picasso said "the chief enemy of creativity is good sense." Consistent validation techniques and re-usable tools greatly reduce resource requirements and repetitive validation routines. We can increase precision in validation of code without sacrificing innovation or exploration. The result? Reliable, re-usable, scalable products positioning our organizations for faster change.

The greatest works of art are produced by gifted people applying well-known techniques. Mastering the art of code validation transforms what some consider a necessary evil into a valued-added necessity—veritable masterpieces!

Introduction

All art is an individual's expression of culture. Cultures differ, so art looks different.

- Henry Glassie

Validation is the process and documentation that provides evidence that specific code or a system will perform in accordance with predetermined requirements.
The challenge of validating code is that not all code looks or behaves the same. We are developing innovative code in a variety of different ways within our organizations to solve specific problems and deliver solutions to our customers. SAS® has several ways of implementing re-usable code including macros, methods, and stored processes. The inherent flexibility of code design impacts validation design because the two are intertwined (Power, 2006). When an efficient macro is written, others want to use it. A well-designed macro can be used in many different situations. A well-designed validation process combined with re-usable software tools can also be used in many situations. In this kind of validation approach, we should assess the level of validation needed, bring together expertise within our organization, and create a standard approach that is used repeatedly.

Preparing for the Gallery

Determine Level of Validation

It's on the strength of observation and reflection that one finds a way. So we must dig and delve unceasingly.

- Claude Monet

We must honestly assess our code and be open to the ideas of others to ensure it is the best product it can be and that, if used by others, will continue to be reliable and scalable. Michelangelo's "David" remains as beautiful a testament to the human spirit as when it was first unveiled 500 years ago. The monumental work has become one of the most enduring artworks in history viewed by thousands of people from all over the world. While our code will not likely be used 500 years from now, the art of validation requires us to examine our code carefully and validate appropriately in order to give it the longest lifespan possible. Additionally, well-designed code will be useful in a broader spectrum than originally intended. Not only will many people view your masterpiece for a long time in

the future, but they will also interpret your work in different ways. The challenge of validation is to prove re-usability of the code in the future as the environment changes and also in different applications other than that for which it was originally intended.

Over-validating can be a time waster and under-validating could produce erroneous results and analysis. Did Michelangelo consider the enormous height of David when it was time to transport him to the Piazza della Signoria in Florence? It took four days and considerable thought and ingenuity on the part of an architect to transport the sculpture from Michelangelo's workshop to its pedestal (King, 2003).

Because we all work with limited resources, we must "right size" validation of our code. The level of validation we perform is dependent upon:

- How frequently our code will be used
- The number of different execution environments
- The number of diverse uses and studies in which our code will be utilized
- The consequences of erroneous results
- The regulatory requirements in different regions where our code will be used
- The complexity of our code

Knowing this information allows us to classify our code and more accurately determine the level of validation. Code designed for a single project, used once and by only one person, is probably less complex than code that is used across projects and by many people. Therefore, the number of test cases is usually smaller for single-use code versus multi-use code. Multi-use code may be executed by another calling program, stored in several common repositories, and used across different platforms; therefore, approach to validation may be more meticulous and demanding. Validation of single-use software proves that a single execution produces correct results; validation of multi-use software proves any execution will produce correct results.

Another important factor influencing right-size decisions is to evaluate the seriousness of incorrect results produced by our code. We must assess the consequences and severity of mistaken outcome in order to determine level of validation. For example, if an error leads to an incorrect evaluation of the safety of a drug, then the level of validation should be higher than if an error leads to an incorrect investigator zip code in a listing.

In addition to these factors regarding the code itself, right-sizing also requires us to understand and interpret legal and regulatory requirements. This is not always easy to do in an environment where there are multiple regulatory agencies and the regulations are deliberately general enough to allow for different interpretations leading to different implementations. Within our own organizations, varying interpretations can lead to difficulties in coming to consensus. These factors may influence us to second-guess what the FDA would say in an inspection leading to "compliance paralysis" as described by Hancock (2005).

Uncertainty can lead to unnecessary levels of validation and this kind of thinking does not allow validation to be done efficiently. To combat "compliance paralysis", Hancock recommends we establish what documentation makes good business sense. When we focus on good business sense as well as on compliance, we can create streamlined processes and eliminate wasted effort.

Complexity of code usually does not lead to a different level of validation, but it can influence the number of test cases. By complexity, we mean code that implements a greater number of function points. As the complexity of code increases, so might the number of test cases. Do not confuse the level of validation performed with the volume of test cases. Even lower levels of validation make take more resources than higher levels of validation because of code complexity. The level of validation is not the only determinant of the amount of resources needed to appropriately validate your code.

The longer you look at an object, the more abstract it becomes, and, ironically, the more real.

- Lucian Freud

Deciding on the right level of validation for your code is similar to artists deciding on the best frame and matting for their work. Choosing the right frame transforms the piece into a powerful experience for the audience; choosing the wrong one can ruin even the finest piece or prevent it from being seen at all! A good frame protects, enhances, and finishes a painting. We want the validation of our code to do the same. How you expect your code to evolve and the consequences of getting it wrong will influence how much effort you put into validation.

What's on your Palette?

Find the Right Resources

If we want to be sure we are making the right validation decision for our code, we might do what an artist would do when a painting is finished: consult the experts! A professional framer knows from experience and instinct, which frames will work best with a particular piece. Like an artist, a good framer has a good eye, understands the elements of contrast, composition, and color.

We cannot expect any one person to have the expertise in all things associated with effective validation. Our commitment to quality during the process should motivate us to collaborate with experts to not only increase our confidence in the product but to build trust from clients who will utilize the product.

I have forced myself to contradict myself in order to avoid conforming to my own taste.

- Marcel Duchamp

Teams are essential to any organization and the process of code validation is no different. There's a myth that to be a truly creative artist, one must work alone. Certainly, code development requires individual concentration, but there comes a time when the help of others is needed to make it public. When a sculptor is ready to have her work shown, she may rely on a representative to find an appropriate forum, needs someone to understand the elements involved in selling the work (if it is

for sale), and requires individuals to help her transport the piece safely to its destination. Adding expertise to your palette both during and after the code is developed can be beneficial in many ways.

Dramatic benefits can be realized from involving representation from the business. We may never fully realize the power of our code until we understand the needs of the business. Involving business intelligence into our validation activities enhances our ability to build code that matters and increases the likelihood of user acceptance.

Consider the following roles when building a validation team:

- Requestor – identifies the need
- Owner – oversees the process
- Analyst – gathers requirements
- Author – designs and writes code
- Test designer – ensures requirements are met
- Peer reviewer – reviews code and executes tests

These roles can provide skills to address:

- business intelligence
- quality insight
- software development proficiency
- testing expertise
- coordination

The onlooker has the last word, and it is always posterity that makes the masterpiece.

- Marcel Duchamp

Those who understand the big picture add tremendous value to our validation efforts. They have a shared objective of moving the organization toward a common goal and can also assist with managing validation timelines, resources, and implementation plans. They need not understand the code to add value in the validation process. Business intelligence not only justifies code creation but ensures it has the necessary support to deliver and maintain it. The analyst role, then, has the challenge of translating this intelligence into distinct and testable requirements.

It is through art, and through art only, that we can realize our perfection.

- Oscar Wilde

Adding a quality representative to your validation team provides advice on a validation strategy based on regulatory and corporate SOP requirements. This includes designing a validation plan, guidance on creating validation deliverables, and how to maintain a validated state for your code. It is important to have a perspective for quality in code engineering as well as quality assurance. Individuals coming from a quality perspective can ensure that we adhere to coding standards, use required validation tools, and document our code properly. This reduces errors and increases longevity of our product. Additionally, quality insight ensures compliance by working with training staff and coordinating inspections.

An artist is not paid for his labor but for his vision.

- James McNeill Whistler

Expertise in information technology and software development lifecycle (SDLC) provides assurance that code executes properly based on requirements. Great artists receive commissions for completing a desired piece. A code author receives requirements for completing desired software. A master artist will use technical and inventive skills to create a work of art that inspires and draws his audience. A master code author will use both technical and inventive skills and tools to create effective and creative code. Well-designed code has a clear link to its requirements and can be tested against those requirements.

I do not seek, I find.

- Pablo Picasso

Testing assures that code is meeting defined requirements thus providing solutions to the business problem; however, it is an area we are tempted to short change. If we succumb to this temptation, we become vulnerable to increased inspection findings and code that may not perform correctly. Skilled test designers can produce test plans that exercise the conditional code logic, inputs, outputs, and execution environments. Individuals creating tests for our code can reveal vulnerabilities where our algorithms begin to fail. Recruiting the help of those with a knack for writing test programs that try to "break" your code is also advantageous (although you may not think so at the time).

The roles discussed above can be thought of as the different paints on your palette—the palette itself can be viewed as the "owner" coordinating the entire validation process. This role provides the integrated planning and coordination necessary to achieve the full benefits of the validation process. Responsible for planning and execution, the owner is accountable for many activities including planning, resourcing, managing timelines, assessing and mitigating risk, producing validation deliverables and ensuring SOP compliance, strategic influencing, communicating, etc.

Paint colors each have their distinct value and an artist recognizes the need for a palette to combine them harmoniously or keep them separated as needed. Roles also have distinct value in a validation process and the owner's expertise in creating a collaborative effort is critical to creating a harmonious product. The owner must also keep certain roles separated (e.g., the code author should not design testing). An effective owner exploits the value of each individual role and combines them into an effective team.

Senge (1990) said that the prominent feature of being part of a great team is the "*meaningfulness* of the experience." Those who've been there acknowledge a feeling of "being part of something larger than themselves, of being connected and generative"; they frequently spend the remainder of their lives searching for a way to recapture that spirit. But he goes on to establish another indisputable benefit: innovative learning organizations exist because of teams with the capacity to discover insights that cannot be attained individually. Team intelligence exceeds the intelligence of individuals on a team

and this develops “extraordinary capacities for coordinated action.”

Maintaining Creativity

Automating Validation

Creativity takes courage.

- Henri Matisse

Standard validation processes do not need to stifle creativity in code design. In truth, it can allow us to unlock creative potential and experience more moments of insight where we see the business, a problem, or an idea in a new way. Standard testing tools, methods, and materials can automate the time-consuming, mechanical steps of validation and need to “reinvent the wheel.” Imagine Picasso having to manufacture his own brushes and paints rather than creating works of art. Art lovers everywhere would suffer! In a like manner, if code authors continue to spend too much time and thought on the mechanics of validation, we lose opportunities to explore and innovate. Standardization can reduce validation time, save money, build morale and job satisfaction, as well as introduce and maintain quality in an organization.

Regulated environments require us to conform to strict policies and procedures. We are tempted to believe that this poses a threat to our creativity yet it can actually work in our favor! If we automate and standardize validation tasks, we do not have to allow regulations to paralyze creativity and spontaneity. Standardization creates a sense of order and offers the reassurance of the familiar—this can free one’s natural creative impulses. Power (2006) notes that “there is a competitive advantage available for those companies who proactively incorporate validation into their broader collaborative vision, and who look for ways to reduce waste and improve efficiency within the validation procedures.” Standardization and creativity must co-exist if organizations are to succeed.

Some examples of potential areas in which we can standardize include:

- Centralized development, testing, and production areas
- Templates for requirements gathering, test design, and validation summary reports
- Techniques for automating repetitive validation routines

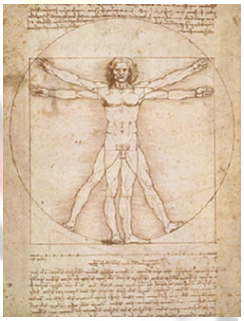
Centralized development, testing, and production areas increase consistency and coherence. A development area provides a place for code authors, analysts, and test designers to begin their work with access to standard templates and tools. Additionally, it allows for collaboration among those involved at this stage of the validation process. The “owner” role (who may be involved in the validation of more than one piece of code) can monitor progress of the validation process in one place. When entering a development area, validation team members will know what to expect and what kind of activities they can complete. For example, there is an understanding that materials in a development arena are subject to change. In a testing area, security is higher and stability of materials there is demanded. And, of course, the production area is the gallery itself with the highest security (motion detectors and all)! An organization that creates centralized and standardized areas for validation reduces redundant effort and ensures the use of consistent tools and processes.

Using the same directory structures and templates within a central area makes it easier to create and find validation materials. Moreover, it opens up possibilities to automate routine validation activities. Automating testing is by far the most significant area of validation where tremendous benefits can be gained. Testing is a repetitive process and automated testing tools can be added to the validation process when it is beneficial.

The execution of all test programs can be automated by writing a script that runs all test programs, sequentially, in a separate SAS session. If each of the test programs report whether a test passes or fails, the script can generate a list of each test program name that was run along with its success or failure.

The test program can determine the success or failure of the test using two methods. The first is proc compare to compare actual output data sets to expected data sets. Proc compare generates a return code so that the test program can determine whether or not to abort the program. In this way, test failures can be reported. The other method is to use a SAS facility to call operating commands like “fc” (file compare). This type of system call also generates a return code allowing the test program to determine success or failure of the test.

The methods above automate test execution. To automate the creation of the test programs, test definition data sets can be populated with the information about each test case (e.g., name of the macro being called with each of its parameter values). A SAS program can read these test definition data sets and write out each test program. Another SAS program can read these same test definition data sets and write a test script document. In other words, we not only have occasions to automate test execution but we can automate the creation of our validation documents as well! The same creativity for writing the software being tested can be used in creating an automated test environment. This permits you to focus on the work of art, rather than on the paints and brushes. The mechanics of repetitive testing, while important, should not shift the focus away from the creation of useful software.



When attempting to create large and complex things, it is a good idea to create a smaller model first of those things from which to learn. Sculptors are known to begin their work in bronze and marble by first creating a smaller model in clay. This allows them to learn about the larger work in an environment that allows for more rapid change and adjustments. Code development can also benefit from this kind of approach. For example, developing SAS macros in a single-use environment allows the author to learn the strengths and weaknesses of their design and to make modifications more easily without multi-use validation overhead. This code development approach can be

viewed as "multiple, single-use testing" and can add value by delivering more bug-free code to formal, multi-use validation. This is not to be confused with validation-in-use because the macro is being formally tested each time it is used. This approach to testing allows the code author to learn and adjust. It is easier to change an 8' clay model of Da Vinci's Horse than to modify the 24' bronze statue.

Creativity is allowing yourself to make mistakes. Art is knowing which ones to keep.

- Scott Adams

It is important to identify those things in the validation process that are obstacles to rapid development so they can be addressed. Automation and standardization, applied correctly, are powerful methods to remove the barriers allowing us the freedom to be inventive.

Summary

An artist cannot fail; it is a success to be one.

- Charles Horton Cooley

Validation, if done well, can reduce expenses, eliminate waste, increase efficiency, and ensure quality. Validation done poorly may restrict a company's ability to be competitive and unable to cope with rapid change. For those that master the art of validation, the benefits are enormous. As long as there are exceptional code authors and useful validation environments, an organization can meet the dual goals of rapid-technology development and compliance to regulations.

Claude Monet's most famous masterpiece might be "Water Lilies." The "Girl With Pearl Earring" would be Rembrandt's. Van Gogh's is "The Starry Night." What constitutes a masterpiece? The simplest of definitions is that it is something that makes you want to look at it, listen to it, or use it over and over; something that represents human creative genius.

Pablo Picasso was once asked which of his paintings was his favorite. He replied, "The next one," meaning that the greatest interest to the artist is the creative process and constructing something that inspires interaction. The code author shares the same motivation as the artist; enjoying the creative process and developing systems that end-users want to use. However, the code author must work beyond the initial creative concept to create the final, polished product. This mechanical, technical work can be enhanced by a superior validation environment.

Validation of code need not need hinder the imagination or obstruct progress; in reality, it fosters the development of innovative thoughts into veritable masterpieces.

References

King, Ross, 2003. *Michelangelo & the Pope's Ceiling, p.3.*

Power, Mike, 2006. *Transforming Validation: How Validation Can Cost Less, Function More Efficiently, and Add Value to the Enterprise.* EnteGreat, Inc., Alabama.

Hancock, Diane, 2005. *The Correct Path to Qualifying and Maintaining Your Infrastructure.* Eli Lilly and Company, Indiana.

Senge, Peter M. 1990. *The Fifth Discipline: The Art & Practice of the Learning Organization.* Doubleday, New York, N.Y.

Acknowledgments

The authors would like to thank Sharon Naylor, Associate Area Quality Consultant, Eli Lilly and Company, for her contributions and review of this paper.

Trademark Notice

Author Contacts

Mindy S. Rodgers
Eli Lilly and Company
Lilly Corporate Center
Drop Code 2233
Indianapolis, IN 46285
317.277.7062
msrodgers@lilly.com

Gregory C. Steffens
Eli Lilly and Company
Lilly Corporate Center
Drop Code 2233
Indianapolis, IN 46285
Phone 317.651.4857
steffensgc@lilly.com