International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2024: 7.101

Computer Software Validation of Design of Experiments Software

Waghchaure Rajeshree Suresh

Mahatma Gandhi Vidyamandir's Pharmacy College, Panchavati, Nashik-422003, India
Email: *waghchaurerajeshreesuresh[at]gmail.com*

Abstract: ***Computer software validation plays a crucial role in ensuring the quality and reliability of pharmaceutical processes, particularly within the framework (QbD). In this context, Design of Experiments (DOE) software is an essential tool used to optimize formulations, processes, and manufacturing systems, ensuring that pharmaceutical products meet desired specifications. This review examines the critical aspects of computer software validation in the pharmaceutical industry, specifically focusing on DOE software used in QbD. It highlights the regulatory requirements, best practices, and methodologies involved in validating software tools, ensuring that they meet stringent standards for accuracy, reliability, and traceability. Additionally, the review discusses the challenges encountered during validation processes, including software documentation, risk assessments, and system integration, while emphasizing the need for continuous monitoring and revalidation in a constantly evolving regulatory landscape. By addressing the intersection of software validation, QbD principles, and DOE software applications, this review aims to provide a comprehensive understanding of how proper validation supports the design, development, and manufacturing of high-quality pharmaceutical products.***

Keywords: Regulatory Compliance, Software Documentation, Risk Assessment, Process Optimization, Software Revalidation

1. Introduction

Computer software validation becomes an essential process to ensure that software used in QbD, such as DOE tools, meets both regulatory standards and industry best practices. Validation ensures that the software functions as intended, produces accurate results, and is capable of operating within the highly regulated pharmaceutical environment. This process involves thorough documentation, rigorous testing, risk assessments, and ongoing monitoring to ensure the software's continued compliance throughout its lifecycle.

This review aims to explore the role of software validation in ensuring the accuracy, reliability, and regulatory compliance of DOE tools used in QbD, with the ultimate goal of supporting the development of safe and effective pharmaceutical products.

2. QbD in pharmaceuticals is significant because it:

- Ensures Consistency and Quality,
- Improves Process Understanding,
- Regulatory Compliance,
- Maintaining Risk Mitigation,
- Cost Efficiency

3. Relevance of Computer Software Validation with QbD:

In the context of QbD, computer software validation becomes a critical element to ensure the integrity and reliability of the tools used in the design and development process. Pharmaceutical companies rely heavily on advanced software tools for various purposes, including Design of Experiments (DOE), process optimization, data analysis, and simulation. These software tools are used to design and analyse experiments, evaluate different formulation or process parameters, and determine the optimal conditions for manufacturing.

4. The relevance of computer software validation with QbD in pharmaceutical field is crucial because of the following reasons:

a) **Regulatory Compliance:** Regulatory agencies like the FDA require that software used in the pharmaceutical industry be validated to ensure it performs as intended and produces accurate results. Validation ensures that the software adheres to the principles of Good Manufacturing Practice (GMP) and complies with regulatory standards.

b) **Accuracy and Reliability of Results:** The results generated by DOE software or other process simulation tools directly impact the development and optimization of pharmaceutical products. Validation ensures that the software provides accurate, reliable, and reproducible results, minimizing the risk of errors during product development.

c) **Traceability and Documentation:** Validated software provides a clear record of all activities, including the design, execution, and analysis of experiments. This documentation is crucial for demonstrating compliance with regulatory standards and for audit purposes. It ensures traceability, which is critical for maintaining the integrity of the QbD processes.

d) **Risk Management:** A validated software tool helps to mitigate risks associated with data manipulation, system failures, and human errors. By ensuring the software functions correctly, pharmaceutical companies can reduce the likelihood of adverse outcomes caused by software-related issues.

e) **Continuous Improvement:** Validation is not a one-time activity but an ongoing process that involves periodic checks and revalidation to ensure continued compliance. This supports QbD's philosophy of continuous improvement in product and process design, allowing for

real-time adjustments and refinements in the development lifecycle.

f) **Support for Process Optimization:** QbD focuses on optimizing the entire process. Validated software tools, especially those used for DOE, enable pharmaceutical companies to efficiently analyze multiple variables and optimize formulation and process conditions to achieve the desired quality outcomes. This contributes to achieving the optimal balance between product quality, cost, and manufacturing efficiency.

5. Meaning of Computer Software Validation in Pharmaceuticals

Computer software validation refers to the process of ensuring that software systems used in the pharmaceutical industry function as intended, meet predefined requirements, and comply with regulatory standards. In the context of pharmaceutical software, validation involves documenting, testing, and confirming that the software performs its tasks accurately, reliably, and consistently throughout its intended lifecycle. This process is essential when using software for critical functions such as data collection, analysis, manufacturing control, regulatory reporting, and decision-making.

6. Regulatory Requirements for Software Validation

a) **Requirements:** A documented software requirements specification provides a baseline for both validation and verification. The software validation process cannot be completed without an established software requirements specification.

b) **Software Life Cycle:** Software validation takes place within the environment of an established software life cycle. The software life cycle contains software engineering tasks and documentation necessary to support the software validation effort. In addition, the software life cycle contains specific verification and validation tasks that are appropriate for the intended use of the software.

c) **Plans:** The software validation process is defined and controlled through the use of a plan. The software validation plan defines "what" is to be accomplished through the software validation effort. Software validation plans are a significant quality system tool. Software validation plans specify areas such as scope, approach, resources, schedules and the types and extent of activities, tasks, and work items.

d) **Procedures:** The software validation process is executed through the use of procedures. The procedures should identify the specific actions or sequence of actions that must be taken to complete individual validation activities, tasks, and work items.

e) **Software validation after a change:** Due to the complexity of software, a seemingly small local change may have a significant global system impact. When any change (even a small change) is made to the software, the validation status of the software needs to be re-established. Whenever software is changed, a validation analysis should be conducted not just for validation of the individual change, but also to determine the extent and impact of that change on the entire software system. Based on this analysis, the software developer should then conduct an appropriate level of software regression testing to show that unchanged but vulnerable portions of the system have not been adversely affected. Design controls and appropriate regression testing provide the confidence that the software is validated after a software change.

f) **Independence of Review:** Validation activities should be conducted using the basic quality assurance precept of "independence of review." Self-validation is extremely difficult. When possible, an independent evaluation is always better, especially for higher risk applications. Some firms contract out for a third-party independent verification and validation, but this solution may not always be feasible. Another approach is to assign internal staff members that are not involved in a particular design or its implementation, but who have sufficient knowledge to evaluate the project and conduct the verification and validation activities. Smaller firms may need to be creative in how tasks are organized and assigned in order to maintain internal independence of review.

7. Activities and Tasks

Software validation is accomplished through a series of activities and tasks that are planned and executed at various stages of the software development life cycle. These tasks may be one time occurrences or may be iterated many times, depending on the life cycle model used and the scope of changes made as the software project progresses.

a) **Software Life Cycle Activities:**

Software developers should establish a software life cycle model that is appropriate for their product and organization. The software life cycle model that is selected should cover the software from its birth to its retirement. Activities in a typical software life cycle model include the following:

- Quality Planning
- System Requirements Definition
- Detailed Software Requirements Specification
- Software Design Specification
- Construction or Coding
- Testing
- Installation
- Operation and Support
- Maintenance
- Retirement

Verification, testing, and other tasks that support software validation occur during each of these activities. A life cycle model organizes these software development activities in various ways and provides a framework for monitoring and controlling the software development project. Several software life cycle models (e.g., waterfall, spiral, rapid prototyping, incremental development, etc.)

b) Typical Tasks supporting Validation:

For each of the software life cycle activities, there are certain "typical" tasks that support a conclusion that the software is validated. However, the specific tasks to be performed, their order of performance, and the iteration and timing of their

performance will be dictated by the specific software life cycle model that is selected and the safety risk associated with the software application. For very low risk applications, certain tasks may not be needed at all.

c) Requirements:

Requirements development includes the identification, analysis, and documentation of information about the device and its intended use. Areas of special importance include allocation of system functions to hardware/ software, operating conditions, user characteristics, potential hazards, and anticipated tasks. In addition, the requirements should state clearly the intended use of the software.

The software requirements specification document should contain a written definition of the software functions. It is not possible to validate software without predetermined and documented software requirements. Typical software requirements specify the following:

- All software system inputs;
- All software system outputs;
- All functions that the software system will perform;
- All performance requirements that the software will meet, (e.g., data throughput, reliability, and timing);
- The definition of all external and user interfaces, as well as any internal software-to-system interfaces;
- How users will interact with the system;
- What constitutes an error and how errors should be handled;
- Required response times;
- The intended operating environment for the software, if this is a design constraint (e.g., hardware platform, operating system);
- All ranges, limits, defaults, and specific values that the software will accept; and
- All safety related requirements, specifications, features, or functions that will be implemented in software.

Software safety requirements are derived from a technical risk management process that is closely integrated with the system requirements development process. Software requirement specifications should identify clearly the potential hazards that can result from a software failure in the system as well as any safety requirements to be implemented in software. The consequences of software failure should be evaluated, along with means of mitigating such failures (e.g., hardware mitigation, defensive programming, etc.). From this analysis, it should be possible to identify the most appropriate measures necessary to prevent harm.

The Quality System regulation requires a mechanism for addressing incomplete, ambiguous, or conflicting requirements. Each requirement (e.g., hardware, software, user, operator interface, and safety) identified in the software requirements specification should be evaluated for accuracy, completeness, consistency, testability, correctness, and clarity. For example, software requirements should be evaluated to verify that:

- There are no internal inconsistencies among requirements;
- All of the performance requirements for the system have been spelled out;
- Fault tolerance, safety, and security requirements are complete and correct;
- Allocation of software functions is accurate and complete;
- Software requirements are appropriate for the system hazards; and
- All requirements are expressed in terms that are measurable or objectively verifiable.

A software requirements traceability analysis should be conducted to trace software requirements to (and from) system requirements and to risk analysis results. In addition to any other analyses and documentation used to verify software requirements, a formal design review is recommended to confirm that requirements are fully specified and appropriate before extensive software design efforts begin.

Requirements can be approved and released incrementally, but care should be taken that interactions and interfaces among software (and hardware) requirements are properly reviewed, analysed, and controlled.

8. Need of Validation:

The level of software validation should align with the risk posed by the automated process. Factors such as process complexity and the manufacturer's reliance on automation for ensuring product safety and effectiveness influence the extent of testing required. Risk analysis and documented requirements help determine the necessary validation scope.

For instance, minimal testing may suffice for an automated milling machine if its output is fully verified before release. In contrast, extensive validation is needed for high-risk systems such as plant-wide electronic record systems, sterilization controllers, or automated test equipment for life-supporting devices.

Commercial software used in quality systems—like spreadsheets, databases, or graphics tools—must be validated based on intended use. Only the functions used and relied upon by the manufacturer need validation. However, high-risk applications should not share an environment with non-validated software functions, even if unused. In such cases, risk mitigation techniques like memory partitioning may be necessary.

9. User Requirements

- The “intended use” of the software or automated equipment; and
- The extent to which the device manufacturer is dependent upon that software or equipment for production of a quality medical device.
- Document requirements for system performance, quality, error handling, startup, shutdown, security, etc.
- identify any safety related functions or features, such as sensors, alarms, interlocks, logical processing steps, or command sequences; and
- Define objective criteria for determining acceptable performance.

The validation must be conducted in accordance with a documented protocol, and the validation results must also be documented. Test cases should be documented that will

exercise the system to challenge its performance against the pre-determined criteria, especially for its most critical parameters. Test cases should address error and alarm conditions, startup, shutdown, all applicable user functions and operator controls, potential operator errors, maximum and minimum ranges of allowed values, and stress conditions applicable to the intended use of the equipment. The test cases should be executed and the results should be recorded and evaluated to determine whether the results support a conclusion that the software is validated for its intended use.

10. Validation of Off-The Shelf Software

Most of the automated equipment and systems used by device manufacturers are supplied by third party vendors and are purchased off-the-shelf (OTS).

The vendor's life cycle documentation, such as testing protocols and results, source code, design specification, and requirements specification, can be useful in establishing that the software has been validated.

11. Key Parameters for DoE Software Validation by QbD Approach:

1) Intended Use & Functionality:

a) **Purpose:** Enlisting the elements which will be supported by the QbD Software.

b) **Functionality Requirements:**
- Experimental design generation (e.g., factorial, response surface, mixture designs)
- Statistical analysis (ANOVA, regression, residuals)
- Visualization tools (contour plots, interaction plots)
- Optimization algorithms

c) **User Role Management:** Permissions for design, analysis, review, export

2) Data Integrity & Security (aligned with 21 CFR Part 11)

a) **Audit Trails:** Tracking who did what, when, and why

b) **Access Control:** Unique user logins, role-based access

c) **Data Retention:** Secure and long-term storage

d) **Data Backup & Recovery:** Tested protocols for data restoration

3) Software Verification & Testing

a) **Installation Qualification (IQ):** Installation steps verified against vendor documentation

b) Operational Qualification (OQ):
- Functionality tested under simulated conditions (with known datasets)
- Test statistical calculations (e.g., p-values, model coefficients) for accuracy

c) **Performance Qualification (PQ):** Real-world pharmaceutical datasets to confirm consistent, valid outputs

4) Statistical Model Validation

a) **Model Accuracy:** Does the software correctly fit models? Validate against known solutions.

b) **Diagnostics Tools:**
- Residuals
- Lack-of-fit testing
- R^2 and adjusted R^2

c) **Model Adequacy Tools:** Cross-validation, prediction errors, leverage analysis

5) Risk-Based Classification (ICH Q9)

a) **Impact Assessment:** Determine if the software influences CPPs/CQAs

b) **GAMP 5 Classification:** Typically Category 4 (Configurable) or Category 3 (Standard)

c) **Mitigation Strategies**: Document how errors or failures are identified and mitigated

6) Traceability Matrix

a) Connect each user requirement with corresponding validation tests.

b) Ensures full traceability of all critical requirements (URs → Test Cases → Results)

7) Documentation

Validation Plan and Report

8) Regulatory Compliance

a) Software operation and analysis

b) Report generation and interpretation

c) Training and access procedures

9) Integration into QbD Lifecycle

a) FDA 21 CFR Part 11

b) EU Annex 11

c) ICH Q8/Q9/Q10/Q11

d) GAMP 5 Guidance

10) Integration into QbD Lifecycle:

a) Design Space Justification: Can outputs be used in regulatory submissions?

b) Technology Transfer: Can models be reused or adapted for scale-up sites?

c) Continuous Improvement: Support for lifecycle management and data re-analysis.

12. Steps for validating a pharmaceutical computer software

Validating a Design of Experiments (DoE) software for use in the pharmaceutical industry especially under a Quality by Design (QbD) approach, requires ensuring that the software reliably supports process understanding, risk management, and control strategies as outlined in ICH guidelines (especially ICH Q8, Q9, Q10, and Q11).

To Validate a DoE Software, one should;

1) Define Intended Use (User Requirements Specification - URS)
- Identify how the software will be used (e.g., design space development, factor screening, and optimization).
- Define specific features and capabilities needed (e.g., regression analysis, ANOVA, response surface modeling).
- Align with QbD principles—how will it support risk identification and process understanding?

2) Risk Assessment (ICH Q9 – Quality Risk Management)
- Assess the software's impact on product quality and patient safety.
- Classify the system (GAMP 5: Category 3 – Non-configurable, or Category 4 – Configurable).
- Identify critical operations like statistical modeling or error calculation that affect design space decisions.

3) Vendor Assessment
- Evaluate software provider's credibility, certifications (e.g., ISO 9001), and audit history.
- Request validation documentation (e.g., IQ/OQ, version history, release notes).
- Review support for regulatory compliance (e.g., FDA 21 CFR Part 11, audit trails).

4) Installation Qualification (IQ)
- Verify that the software is installed correctly in the intended environment.
- Check OS compatibility, file integrity, user access setup, and dependencies.

5) Operational Qualification (OQ)
- Confirm that the software performs all functions correctly.
- Test standard functions: model fitting, residual analysis, data import/export, simulation tools.
- Use both typical and boundary-case datasets.

6) Performance Qualification (PQ)
- Demonstrate consistent performance under real-world usage.
- Run real or mock experimental designs relevant to pharmaceutical processes.
- Evaluate if software reliably identifies critical process parameters (CPPs) and quality attributes (CQAs).

7) Traceability Matrix
- Link URS to test cases to show complete coverage.
- Ensure all critical requirements are verified and validated.

8) Documentation and SOPs
- Develop SOPs for software use, maintenance, change control, and training.
- Archive validation protocols and reports (IQ, OQ, PQ).
- Include software version control, backup/recovery procedures, and periodic review schedules.

9) Data Integrity (21 CFR Part 11 Compliance)
- Ensure all user actions are logged securely.
- Confirm data integrity—no unauthorized changes, and all changes are traceable.

10)Change Control and Revalidation
- Establishing a process for managing updates and upgrades.
- Revalidate when functionality changes or regulatory requirements evolve.

13. Conclusion

Validating Design of Experiments (DoE) software within the pharmaceutical industry using a Quality by Design (QbD) approach is critical to ensuring data integrity, regulatory compliance, and robust process understanding. By applying principles from ICH Q8–Q11 and adhering to GAMP 5 and 21 CFR Part 11 guidelines.

The validation process must be risk-based, comprehensive, and well-documented, covering installation, operational performance, statistical accuracy, and secure data handling. Furthermore, integration of the software into the broader QbD lifecycle from early development to commercial manufacturing supports consistent decision-making, enhances regulatory confidence, and facilitates continuous improvement.

Ultimately, validated DoE software becomes a cornerstone in the QbD framework, enabling scientifically sound, data-driven decisions that uphold product quality.

References

[1] US Food and Drug Administration. General principles of software validation; final guidance for industry and FDA staff. Center for Devices and Radiological Health, Tech. Rep. 2002 Jan.

[2] Computer software validation in pharmaceuticals [Internet]. Vols. 1–1, Asian Journal of Pharmaceutical Sciences and Clinical Research. 2011 p. 27–39. Available from: https://www.researchgate.net/profile/Atul-Bendale/publication/215485194_Computer_Software_Validation_In_Pharmaceuticals/links/02e7e51c5c37b8e99c000000/Computer-Software-Validation-In-Pharmaceuticals.pdf

[3] INTERNATIONAL CONFERENCE ON HARMONISATION OF TECHNICAL REQUIREMENTS FOR REGISTRATION OF PHARMACEUTICALS FOR HUMAN USE. (2009). PHARMACEUTICAL DEVELOPMENT. In INTERNATIONAL CONFERENCE ON HARMONISATION OF TECHNICAL REQUIREMENTS FOR REGISTRATION OF PHARMACEUTICALS FOR HUMAN USE. https://database.ich.org/sites/default/files/Q8_R2_Guideline.pdf.

[4] International Conference on Harmonisation of Technical Requirements for Registration of Pharmaceuticals for Human Use. (2005). ICH Harmonised Tripartite Guideline Q9: Quality Risk Management. https://database.ich.org/sites/default/files/Q9_Guideline.pdf

[5] INTERNATIONAL CONFERENCE ON HARMONISATION OF TECHNICAL REQUIREMENTS FOR REGISTRATION OF PHARMACEUTICALS FOR HUMAN USE. (2008). PHARMACEUTICAL QUALITY SYSTEM. In ICH Harmonised Tripartite Guideline. https://database.ich.org/sites/default/files/Q10%20Guideline.pdf

[6] ISPE. (2008). GAMP 5: A Risk-Based Approach to Compliant GXP Computerized Systems.

[7] Fukuda, I. M., Pinto, C. F. F., Moreira, C. D. S., Saviano, A. M., & Lourenço, F. R. (2018). Design of

Experiments (DoE) applied to Pharmaceutical and Analytical Quality by Design (QbD). Brazilian Journal of Pharmaceutical Sciences, 54(spe). https://doi.org/10.1590/s2175-97902018000001006
[8] Katre S, Jain N: Qualification and computer system validation of pharmaceutical instrument: critical quality attributes in pharmaceutical industry. Int J Pharm Sci Res. 2020, 11:5182-91.
[9] Yogesh P, Kamlesh M, Mohini B, et al.: Computer system validation: a review. World J Pharm Res. 2015,4:444-54.
[10] Ray B, Khan AA, Rahman A, et al.: An overview on pharmaceutical software. Int Res J Modern Eng Technol Sci. 2023, 5:477-92.
[11] Singh A, Singour P, Singh P: Computer system validation in the perspective of the pharmaceutical industry. J Drug Deliv Ther. 2018, 8:359-65.
[12] Bendale A, Patel N, Jadhav AG, et al.: Computer software validation in pharmaceuticals. Asian J Pharm Clin Res. 2011, 1:27-39.
[13] Samson Y, Kereon A, Basel: Computerized system validation: regulatory compliance and process safety in the pharmaceutical industry. Conference ERTS'06. 2006, 1:1-7.
[14] Bhaskar R, Ola M, Gosavi H: Review on: computer system validation in pharma industry. Int J Pharm Res Appl. 2023, 8:1759-64.
[15] Hesham AM: Computerized systems validation (CSV) in biopharmaceutical industries. Open Access J Pharm Res. 2020, 4:1-15. 10.23880/oajpr-16000219
[16] Pawar R, Kabra S, Bhushan B: Computer software validation: importance in pharma industry . Int J Pharm Res Appl. 2023, 8:620-6.
[17] Pedro F, Veiga F, Mascarenhas-Melo F: Impact of GAMP 5, data integrity and QbD on quality assurance in the pharmaceutical industry: how obvious is it?. Drug Discov Today. 2023, 28:103759. 10.1016/j.drudis.2023.103759
[18] Uzzaman S: Computer systems validation: a systems engineering approach. Official J ISPE. 2003, 23:1-10.
[19] Dilip Patil R, Pansare JJ: Computer system validation in the perspective of the medical field introduction. Int J Pharm Pharm Sci. 2022, 23:329-6.
[20] Savitha S, Kathiresan K: Computer system validation: a review. Int J Biol Pharm Allied Sci. 2022, 11:5256-66. 10.31032/ijbpas/2022/11.11.6567
[21] Rusjan B: Computer system validation: example of quality management system design and of process implementation. J Contemp Manag Res. 2020, 25:1-23. 10.30924/mjcmi.25.2.1
[22] Patel H V, Yogi RR, Narang E: A review on computer aided instrument validation. J Chem Pharm Res. 2011, 3:134-43.
[23] Jadhav SV, Waghchaure SS, Chemate SZ: Computer system validation in pharmaceutical industry. Int J Creat Res Thoughts. 2021, 9:2320-882.
[24] Vs T, Ks, D'souza P: Computerized system validation: introduction implementation and regulations – a review. Int J Pharm Res Scholars. 2014, 3:122-31.
[25] Bharkatiya M, Jain K, Agarwal P: A review on pharmaceutical validation and its implications. Int J Pharm Biol Sci. 2018, 8:117-26.
[26] Schönberger M, Vasiljeva T: Towards computer system validation: an overview and evaluation of existing procedures. J Innov Manag Small Medium Enterprises. 2018, 1-15. 10.5171/2018.512744
[27] Sai D: Overview of regulations on GMP computerized systems. World J Pharm Res. 2019, 8:618-29.
[28] Hooda A, Nanda A, Jain M, Kumar V, Rathee P. Optimization and evaluation of gastroretentive ranitidine HCl microspheres by using design expert software. International Journal of Biological Macromolecules [Internet]. 2012 Aug 3;51(5):691–700. Available from: https://www.sciencedirect.com/science/article/abs/pii/S0141813012003030#preview-section-references

Author Profile

Waghchaure Rajeshree Suresh Studying Master of Pharmacy in Quality Assuance specification from Mahatma Gandhi Vidyamandir's Pharmacy College, Panchavati, Nashik-422003 under Savitribai Phule Pune University, Pune, Maharashtra, since 2023-2025.