BioPhorum
Operations Group
Connect · Collaborate · Accelerate

GUIDANCE ON THE USE OF AGILE IN A GxP ENVIRONMENT

Authors

Roche
Horst Froede
Javier Hernandez

BMS
Jennifer McCarty

Eli Lilly
Amy Schmid
Nick Miller

BioPhorum Operations Group
Steve Atkinson

Contributors

This document was developed through a collaboration under the auspices of BioPhorum IT Group. The contributing members were:

Alexion
Christopher White
Michael Schofield

Amgen
Joe Santana

Biogen
Mike Stroz

Bristol-Myers Squibb
Aidan Desmond
Claudia Yamamoto
Jennifer McCarty
Randy Bugge
Rekha Alaguchellappan

Eli Lilly
Amy Schmid
Anita Morrison
Nick Miller

GSK
Fred Baluyot

Ipsen
Audrey Godimus

Merck KGaA, Darmstadt, Germany
Hufrish Sirohi
Jennifer Baughman

Regeneron
Declan McDonnell

Roche
Horst Froede
Javier Hernandez

Shire
David Lake
Jeet Sarkar

Contents

1.0 Executive summary

The biopharmaceutical industry is challenged to continually deliver and maintain products in a cost-effective way while retaining regulatory compliance. An Agile approach to software development supports the need for an effective and efficient business. However, use of Agile in a GxP environment has been limited due to the perceived regulatory risk. This guidance document provides an approach that ensures delivery of software solutions while at the same time maintaining regulatory compliance.

While this guidance documents details the approach proposed it is clear that key prerequisites need to be in place for Agile to be successfully adopted, including senior management support and engagement, the use of computerized systems and tools in the management and control of the required deliverables (including electronic records with electronic signatures and audit trails), and the full support and engagement of the quality team.

The approach detailed below retains the key traditional stages of the validation plan and validation report, while adapting the design, build and test stages to provide an Agile approach.

The guidance document is based on the delivery of enough functionality for the system to be usable (implementation phase), with further functionality being delivered via change requests (operations phase). Further details of how this would work in reality are provided.

Using the guidance can enable companies to implement software systems in a GxP environment while delivering to the business the benefits of cost, speed and quality, and at the same time retaining full regulatory compliance.

2.0 Introduction

Background

Biopharmaceutical operations (e.g. manufacturing, clinical research) continue to rely heavily on technology. Opportunities for growth, new product lines and new lines of business often depend on getting to market first, so any software solutions that support these activities must be implemented quickly to help achieve this goal. Following an Agile software development methodology is one effective way to accomplish this.

This guidance document is intended to clearly position the use of an Agile software development methodology as a means to deliver the business imperatives while meeting the regulatory guidelines. To deliver both the benefits of speedy implementations while retaining regulatory compliance there is a need to adapt Agile methodologies where appropriate. The specific way in which the computer system validation (CSV) requirements are met when using an Agile methodology will need to differ from traditional methods.

A key success criterion when using an Agile methodology for a regulated computer system (e.g. GMP, GCP) is to partner with the quality organization when developing the specific approach to meeting the computer system validation specifications. While this guidance document will use terminology specific to the Agile Scrum methodology, the principles and best practices are applicable to any Agile approach. This guidance document recognizes that every project is different, based on intended use of the system, type of Agile methodology used and complexity, and there may be practices other than those described here that may work better based on those types of factors.

Determining what CSV requirements apply and the extent to which they apply (sometimes known as right-sizing CSV) to a particular system is primarily based on the intended use of the computer system.

Purpose

The purpose of this document is to provide guidance on the use of an Agile approach in the development of software in a GxP environment.

The approach described in this document will enable the use of Agile methods ensuring delivery of the benefits of Agile while providing a validated solution.

Audience

This document is intended for IT professionals and business partners interested in applying Agile Scrum concepts and methodologies to software development and software configuration efforts within a regulated environment.

Agile definition

Our definition of Agile is based on and derived from the 12 principles behind the Agile Manifesto (see Appendix A).

Our Agile approach includes a paradigm shift away from the document-centered waterfall development to a user and computer system-centered approach, moving from formal approved documents (electronic or paper) to mostly electronic documentation (set of tightly controlled records that will replace the documents). Instead of developing a wish list of high-level requirements we are going to investigate the users intended use and the computer system capabilities. Based on both, we define detailed user stories, which describe how the computer system can be used to meet the users intended use.

Being Agile:

- put the customer/user and their intended use into the center of the computer system development
- foster strong interaction between business users and IT staff, wherever possible in a face-to-face (or real-time) communication
- investigate the computer system to be implemented (online, with hands on the keyboard) and derive the user stories from it to satisfy the users intended use
- divide the development work into small pieces, which can by managed more easily (sprints)
- get immediate feedback from the users in review sessions, where they work online with the system (sprint review)
- include changes even at a late stage of the development to increase user satisfaction or to fulfill new or changed regulatory requirements
- involve quality/validation and compliance representatives into the interactive development of user stories and oversight sprint executions.

Prerequisites for success

The Agile approach that is detailed in this document requires a number of prerequisites to be in place for it to fully deliver its benefits while delivering a validated system. While an Agile approach can be used successfully without all these key elements in place, the full benefits will not be achieved:

- senior management sponsorship and support
- implementation and use of validated supporting computerized systems (e.g. ticketing and testing, code and release management)
- direct and permanent involvement of quality, compliance or validation on the Agile team
- quality organization thinking alignment with the Agile approach.

Agile approach description

The Agile approach aggregates the 'Agile software development approach' with the traditional validation stages: validation planning; design, build and test; and validation reporting. Whereas the validation planning and validation reporting stages remain unchanged, the design, build and test stage will be adapted by using an Agile software development approach.

Our objective is to deliver with each sprint a shippable software package. However, we consider that for the implementation of a comprehensive system, such as an enterprise resource planning (ERP), manufacturing execution system (MES) or other global enterprise solutions, there may be a need to build several 'sprint units' into the first release. This is because the first release has to give the user enough functionality to start using the new system. Therefore, we distinguish between an implementation phase and an operation phase.

The purpose of the implementation phase is to deliver the first release. In the operations phase you can enhance the system with new functionality via change requests. For the design, build and test stage this does not have any impact. However, the validation planning and validation reporting stages may have some differences compared to the implementation phase. Therefore, they are substituted by the change planning and change reporting stages.

Usage of appropriate supporting system tools/applications

The usage of appropriate supporting tools made up of a minimum of validated supporting computer systems intended to manage the required electronic supporting documentation/records is highly recommended. It achieves more flexibility and speed in the management and control of the required deliverables. These deliverables do not need to be paper or electronic documents, they can also be electronic records in a computer system with the correct electronic signatures and audit trail. This opens the way to increased paperless validation. A prerequisite for this is that all involved subject matter experts (SMEs)—business, IT, quality assurance, validation, etc.—are familiar and qualified with these computer systems, and the workflows are defined in efficient ways to reduce the time for approvals collection. Examples for the usage of some computer systems are given in the text below.

The following list is a recommendation of the supporting computer systems and required records that will make this Agile approach efficient and pragmatic:

Required supporting computer systems: those managing validation support/evidences/records that cannot be managed in paper or electronic documents for applying this methodology in an efficient way:

- Lifecycle management systems, intended to cover:
  - system backlog and user stories (system requirement and system functional specifications)
  - categorization and extra data of user stories
  - design specifications
  - code and releases/versions.
- Testing/verification management computer systems, intended to cover:
  - test cases/test scenarios/use cases
  - test executions and evidence
  - installation verification protocols and executions
  - defects and defect resolution tracking
  - traceability: user stories through design, implementation, testing and test results.
- Ticketing computer systems, intended to cover:
  - inventory and basic configuration management
  - change control and management
  - incident and problem management.
- Disaster recovery management computer systems, intended to cover:
  - software and data backup and restore activities
  - disaster recovery scenarios.

Recommended supporting computer systems: those managing validation support/evidences/records, that provide a significant advantage if managed as records instead of in formal paper or electronic documents:

- Training management computer systems, intended to cover:
  - training material development and version control
  - training execution and records management
- Electronic document management computer systems (EDMS), intended to cover the 'documents' that have to be created (e.g. validation plan and report).

Desirable supporting computer systems: those managing validation support/evidences/records that could be 'manually' managed and associated to documents:

- Systems architecture management computer systems, intended to cover:
  - detailed system architecture and components relationships/dependencies
  - overall system landscape and description.
- Process and procedures management computer systems, intended to develop, maintain and formally issue valid operational processes, procedures and instructions:
  - SOPs (standard operation procedures)
  - TOPs (technical operation procedures)
  - user and technical instructions.

Note: if these kinds of computer systems are not available this information should be managed in an EDMS.

3.0 Roles and responsibilities

There are a number of roles required to support an Agile approach. The key responsibilities of various roles are outlined below, including some considerations specific to Agile approaches in a GxP regulated environment:

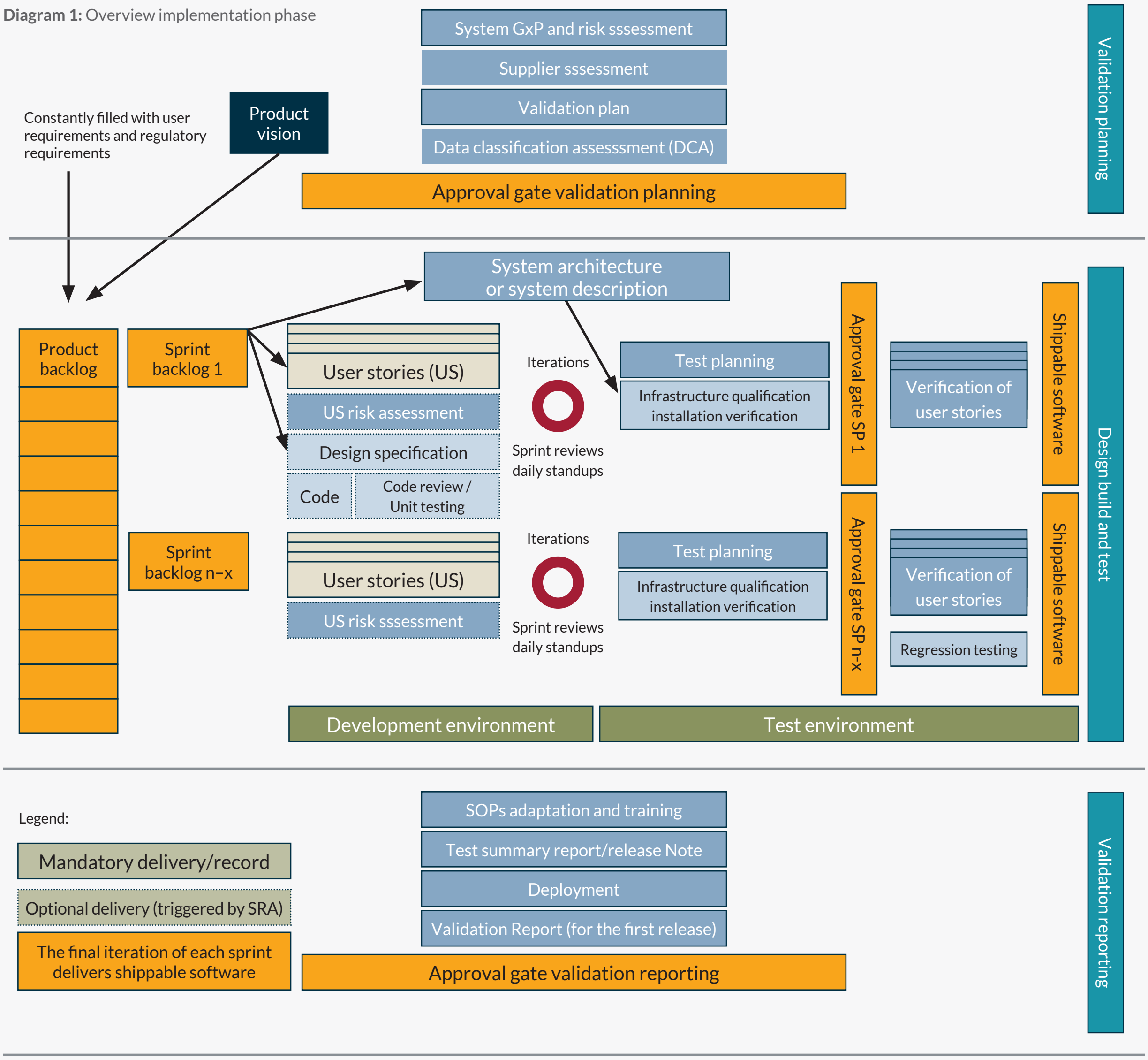
Role	Responsibilities	Considerations
Product Owner	• Define the features of the product • Decide on release • Prioritize features • Adjust features and prioritize every sprint, as needed • Accept or reject work results • Clearly express product backlog items • Ensuring that the product backlog is visible, transparent and clear to all, and determines what the team will work on • Ensuring the team understands items in the product backlog as well as required validation activities/deliverables to the level needed	• If the process owner and product owner are not the same individual, the product owner should be in frequent communication with the system owner • If the Product Owner is not the process owner, a delegation of authority should be considered so that the product owner(s) can approve validation deliverables (see system owner roles and responsibilities)
Scrum Master	• Ensures that the team is fully functional and productive and removes barriers • Enables close cooperation across all roles and functions • Shields the team from external interferences • Helps those outside the team interface with the team properly	• Liaison with QA unit to help ensure that appropriate validation activities and deliverables required for a release are integrated into the development process
Team	• Cross-functional, including the multiple skills (but ideally not the job titles) of architects, software engineers, developers, testers, user-experience designers, etc. necessary to create a product • Membership should change only between sprints • Delivers a potentially releasable increment of 'done' product at the end of each sprint • Individual team members may have specialized skills and areas of focus, but accountability belongs to the team as a whole • Is responsible for the creation of the validation plan • Ensures that traceability between user stories and test cases is maintained	• Although distinct roles can be defined, the intent is that a team member has more than one skill • Validation is included within the team, but can still be organized separately
Process Owner	• The person ultimately responsible for the business process or processes being managed • Single point of contact (POC) for any business decisions or escalations • Along with the Quality Assurance unit, prioritizes and assigns criticality to user stories/requirements • Approves key deliverables as required policy and procedure • Maintains relationship with end-users to understand and influence proper use of the system and with key IT support personnel • Code reviews and testing are considered responsibilities of the team, but care should be taken to insure a separation of duties to avoid a conflict	• May or may not be the product owner but should be providing frequent feedback • The process owner should consider a delegation of authority to the product owner(s) if he/she is not also the product owner

Role	Responsibilities	Considerations
System Owner	• The person ultimately responsible for the availability, and support and maintenance, of a system and for the security of the data residing on that system • Single POC for technical aspects of project and any technical escalations • Approves key deliverables as required by policy and procedure • Evaluates and maintains relationship with suppliers (both those providing the software and any services) • Maintains relationship with key business roles (i.e. system owner, power users) • Oversees unit, integration and system testing to ensure system meets requirements	• May or may not be an Agile team member but should be providing frequent feedback • The system custodian should consider a delegation of authority to a team member if he/she is not a team member
Quality Assurance Unit	• Reviews and approves (when applicable) the GxP and system risk assessment, as required due to the updates to the product backlog or intended use (when new needs are identified and included to be developed) • Participates in product backlog grooming by ensuring business and regulatory requirements have been identified in the product backlog • Provides QA oversight and advice on the activities performed by the team • Monitors to ensure validation deliverables are created and approve • Participates in sprint planning, ensuring quality activities and deliverables are appropriately integrated into the sprint • Participates in the appropriate team status meetings where quality is discussed • Approves the CSV plan, including the CSV Agile strategy described in this guideline • When a sprint was categorized as having GxP impact, review and approve or reject the incremental (per sprint) validation report	• This role is an extended team member (not a core team member) and exists in a separate organizational unit from the business (process owner) and IT.

4.0 IMPLEMENTATION PHASE

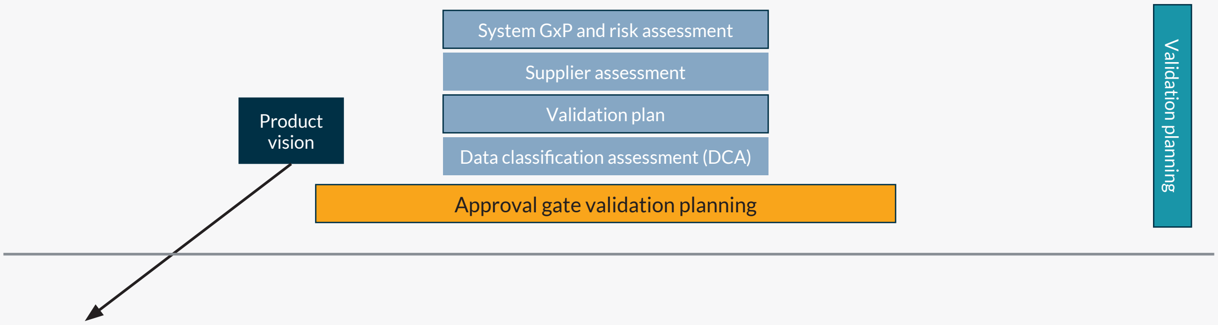
We will describe here the task/activities and deliverables to complement the standard scrum methodology when implementing a GxP system.

Diagram 1: Overview implementation phase


System GxP and risk sssessment
Supplier sssessment
Validation plan
Data classification assesssment (DCA)
Approval gate validation planning
Validation planning
Constantly filled with user requirements and regulatory requirements
Product vision
System architecture or system description
Product backlog
Sprint backlog 1
User stories (US)
US risk assessment
Design specification
Code
Code review / Unit testing
Iterations
Sprint reviews daily standups
Test planning
Infrastructure qualification installation verification
Approval gate SP 1
Verification of user stories
Shippable software
Sprint backlog n–x
User stories (US)
US risk sssessment
Iterations
Sprint reviews daily standups
Test planning
Infrastructure qualification installation verification
Approval gate SP n-x
Verification of user stories
Regression testing
Shippable software
Design build and test
Development environment
Test environment
Legend:
Mandatory delivery/record
Optional delivery (triggered by SRA)
The final iteration of each sprint delivers shippable software
SOPs adaptation and training
Test summary report/release Note
Deployment
Validation Report (for the first release)
Approval gate validation reporting
Validation reporting

5.0 VALIDATION PLANNING STAGE

Diagram 2: Implementation phase – validation planning


System GxP and risk assessment
Supplier assessment
Validation plan
Data classification assessment (DCA)
Product vision
Approval gate validation planning
Validation planning

Product vision

The product vision gives a high level introduction into the system, which should be delivered.

System GxP and risk assessment (SRA)

The SRA is the foundation for a risk-based validation approach. It identifies the risks of the system in various areas by analyzing and reviewing the possible threats based on the criticality of the usage, processes, data and technology the system manages.

The SRA has to consider the impact on the end-user, compliance (laws and regulation) and the enterprise business identifying specific risks values per risk domain to allow the definition of domain specific mitigating actions.

Data classification assessment (DCA)

The purpose of the DCA is to categorize the processed data assets according to its sensitivity (e.g., impact of applicable laws and regulations), identifying the criticality of the data to be able to define appropriate technical data integrity and security controls, as well as legal controls.

Supplier assessment

If a bought software package or configurable off-the-shelf application (COTS) should be used to build the system, the supplier of the application has to be assessed. If other third-party service providers contribute to the delivery and/or support of the system, they also have to be assessed.

This process should be triggered as soon as possible once the supplier(s) or potential suppliers have been identified, and can be run in parallel during the system implementation. It is important to assess the supplier before the design stage to be able to identify and fix any significant risks/gaps related to the suppliers.

Validation plan

The validation plan gives an overview of the system to be delivered. It should cover the planning and strategy for the whole system (as far as it is known at this stage). It defines the detailed validation scope, intended use, task/activities and the roles and responsibilities to carry them out.

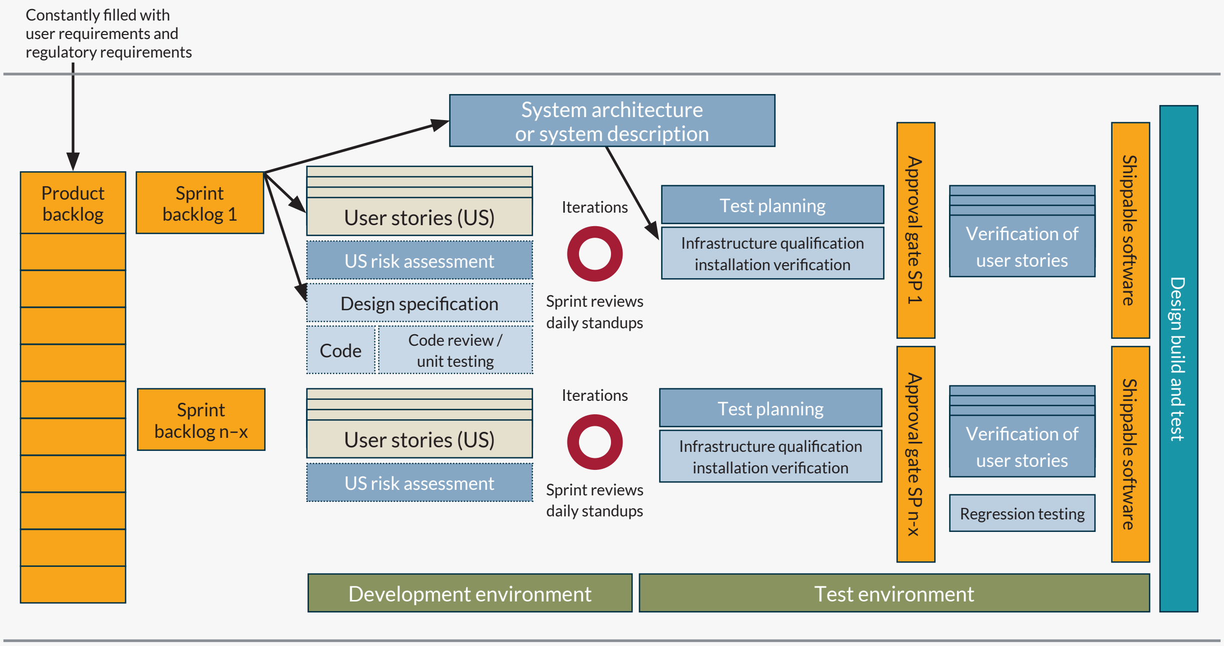
When performing validations using this Agile strategy, the validation plan must clearly state/include the list of supporting systems used to manage the required validation supporting records/evidences (Agile validation environment/ecosystem).

Approval gate validation planning

In order to pass this gate the applicable roles must confirm that SRA, DCA and validation plan are approved and the supporting environment is defined and ready.

6.0 DEFINE BUILD AND TEST STAGE

Diagram 3: Implementation phase – design build and test


Constantly filled with user requirements and regulatory requirements
Product backlog
Sprint backlog 1
Sprint backlog n–x
System architecture or system description
User stories (US)
US risk assessment
Design specification
Code
Code review / unit testing
Iterations
Sprint reviews daily standups
Test planning
Infrastructure qualification installation verification
Approval gate SP 1
Verification of user stories
Shippable software
User stories (US)
US risk assessment
Iterations
Sprint reviews daily standups
Test planning
Infrastructure qualification installation verification
Approval gate SP n-x
Verification of user stories
Regression testing
Shippable software
Design build and test
Development environment
Test environment

Product backlog

The product backlog is constantly filled with high-level user stories, features and regulatory requirements. All are assessed, including GxP relevance and impact (performed by QA), evaluated and prioritized (from a risk point of view) before a new sprint is started. Based on predefined selection criteria, a set of backlog stories is bundled into a sprint. The product backlog does not need to be a document. It can be composed out of records for each entry in an appropriate supporting computer system.

Sprint execution

For each sprint, a quality/compliance or validation representative must be part of the team, providing direct advice and oversight on the validation-related activities and needs.

For each sprint, an associated change control (change record) will be opened to collect all relevant information and references and to gather the required 'approvals to release'.

Sprint backlog

Several features out of the product backlog will be put into a sprint backlog. The scope of the sprint backlog will be developed via various iterations to a shippable software package.

System architecture or system description

Based on the technical requirements, the system architecture will be developed. It describes the technical set-up of the system landscape with the required components (network, hardware, storage, operating system, database etc.)

Detailed user stories

Based on the high-level user stories and needs, detailed user stories have to be developed. These user stories describe the user's interactions with the system. The detailed roles of the actors and the business rules that have to be applied are defined.

With this, segregation of duties, compliance to predicate rules, audit trail review activities to achieve data integrity requirements, the system behavior in case of failure or invalid input and a lot of other scenarios can be covered.

The user stories define the intended use of the system on a very detailed level. The user stories do not need to be defined in a paper or electronic document. They can be documented as records in a supporting computer system (please see section "Verification of user stories/regression testing") and may be viewed as the equivalent of system specifications. The definition of such detailed user stories may look cumbersome, but the content can be re-used and leveraged in the training material, the SOPs and in the knowledge management.

User story risk assessment

If a more detailed risk assessment is required by your SRA, the user stories have to be assessed. The risk assessment should focus on patient safety, product quality and data integrity. The risk assessment can be used to scale testing and/or to define further migration actions such as procedural controls, technical controls, security controls or special training needs. It could be done in a testing tool for each user story.

If the information, which is required for this risk assessment is already available at the time the product backlog is created or updated, this risk assessment could also be conducted for each product backlog entry. In this case, it should be documented in the tool that is used to store the product backlog records.

Design specification

In case an interface or add-on development is necessary, the technical design for that will be described in a 'design specification' deliverable (document or record in a tool).

Code

Based on the design specification, the code will be developed.

Code review/unit testing

White-box testing has to be performed to verify the code and the included objects, modules, etc. This can either be conducted through a formal code review or by unit testing with an appropriate testing tool. The objective of this task is to ensure that the coding/programming standards have been applied and no dead code is delivered. With a testing tool you can evaluate the test coverage of the lines of code, the number of branches and other objects and parameters of the testing. Concepts such as continuous-integration tools, code merger, static and dynamic code tester and version control computer system have to be taken into consideration. These aspects are very important for systems delivered based on bespoke software. For COTS and software as a service (SAAS)-based solutions you are dependent on the workbench software delivered by the supplier.

Iterations/sprint reviews/daily standups

Via various interactive verification techniques the developed system will be constantly challenged by the users to ensure that the development goes in the right direction. Details are dependent on the Agile approach that will finally be used.

Test planning

The test planning will define which user stories and defects from pervious cycles are going into the next formal testing cycle. The need for regression testing of previously tested user stories has to be evaluated by applicable roles and based on the GxP impact of the functions to be tested. If the development of new user stories could have a 'site impact' (indirect impact) on already tested user stories, then these user stories have to be retested. The test planning does not need to be documented in a paper or electronic document. It could also be a 'test set' in the testing supporting computer system that contains the user stories that have to be executed. The test set can be approved in the testing supporting computer systems by the responsible roles. General information required for test execution such as roles and responsibilities, defect/error handling, acceptance criteria and others could be defined in a test strategy that is test-cycle independent and valid for the whole system live cycle (it could be even part of the overarching validation plan).

Infrastructure qualification/installation verification

If any change to the infrastructure had to be done for the sprint, the qualification of the infrastructure has to be performed. If a new application or software version has to be installed before the next testing cycle can be performed, the installation verification protocol and execution has to be documented.

Approval gate sprint

In order to pass this gate, the above-mentioned deliverables (documents or records) have to be available and controlled. The applicable roles must approve (using the change control) the readiness and compliance of the sprint to be verified.

Verification of user stories/regression testing

If the user stories are detailed enough, it may not be necessary to define separate test cases. It could be sufficient to define the expected results for the steps that have to be verified by actual results and screenshots. To gain efficiency, the user stories can be defined as records in a test supporting computer system, including the verification steps. With this you can profit from the fact that the user stories are (almost) equal to the test cases.

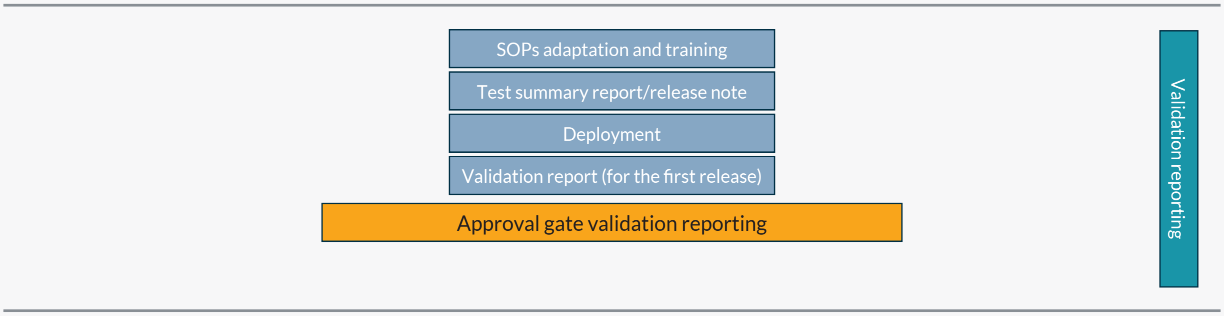
In the regression testing, the site-impacted user stories have to be verified again. Depending on the complexity of the system, the effort for regression testing could become huge. One option to reduce that burden is test automation. If it is possible to automate the verification of the user stories, the effort for regression testing can be minimized in the long run. The automation of testing will offer the opportunity for continuous validation. It is also helpful if you like to leverage the concept of 'DevOps' in the "operation phase".

Shippable software

With the last iteration of the define, build and test stage, we have verified a shippable software package. All these shippable software packages will be bundled together into the first release. This first release will be deployed in the validation and reporting stage.

7.0 VALIDATION AND REPORTING STAGE

Diagram 4: Implementation phase – validation reporting


SOPs adaptation and training
Test summary report/release note
Deployment
Validation report (for the first release)
Approval gate validation reporting
Validation reporting

SOPs adaption and training

Existing SOPs may have to be adapted and new SOPs may have to be developed. This can be applicable for the business and IT areas. Training on updated and new SOPs has to be conducted.

Test summary report/release note

The test summary report will summarize the testing outcome and any open defect will be assessed and classified. The release note will give an overview of the newly available features. When using an appropriate testing tool, the test report may be generated from the tool. This could also be possible for the release note.

Deployment

The deployment will ensure a seamless installation of the new software package on the productive environment and handover to the business users and operation team(s).

Validation report

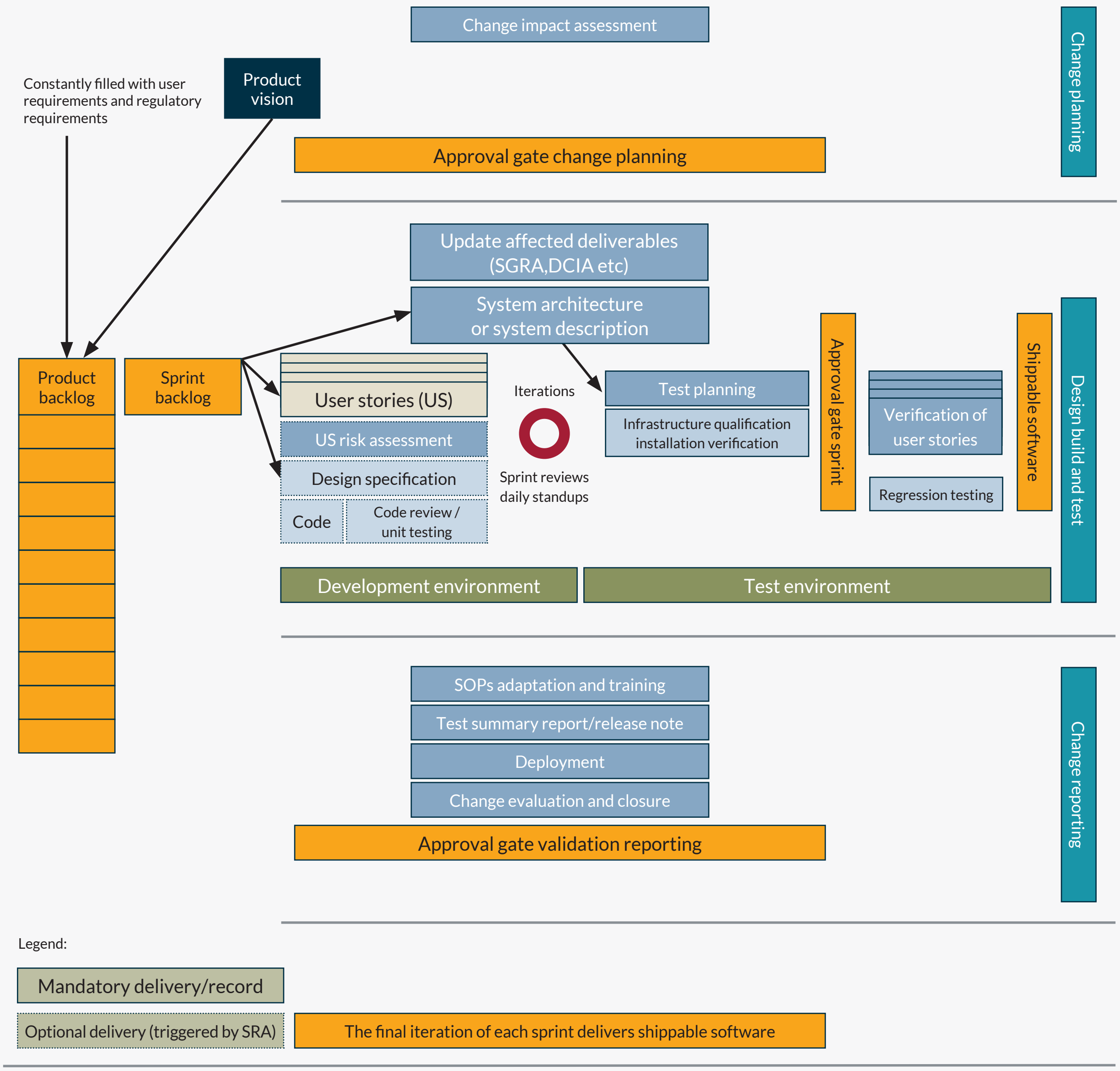
The validation report will summarize the outcome of the validation activities and assess and classify any outstanding items.

Approval gate validation reporting

In order to pass this gate, the above-mentioned deliverables (documents or records) have to be approved.

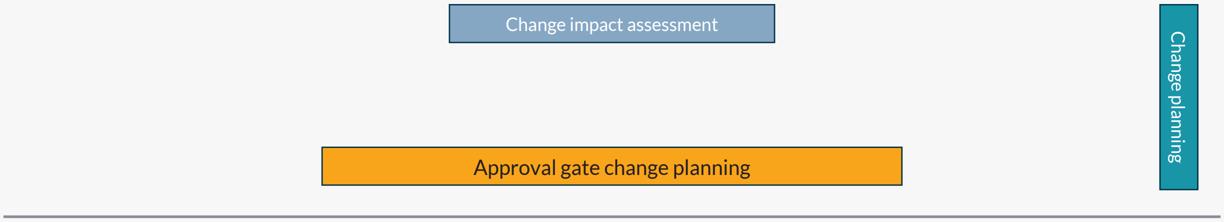
8.0 OPERATION PHASE

Diagram 5: Overview operation phase


Change impact assessment
Change planning
Constantly filled with user requirements and regulatory requirements
Product vision
Approval gate change planning
Update affected deliverables (SGRA,DCIA etc)
System architecture or system description
Product backlog
Sprint backlog
User stories (US)
US risk assessment
Design specification
Code
Code review / unit testing
Iterations
Sprint reviews daily standups
Test planning
Infrastructure qualification installation verification
Approval gate sprint
Verification of user stories
Regression testing
Shippable software
Design build and test
Development environment
Test environment
SOPs adaptation and training
Test summary report/release note
Deployment
Change evaluation and closure
Approval gate validation reporting
Change reporting
Legend:
Mandatory delivery/record
Optional delivery (triggered by SRA)
The final iteration of each sprint delivers shippable software

9.0 CHANGE PLANNING STAGE

Diagram 6: Operation phase – change planning



The change planning is typically done in a ticketing tool. The objective of this task is to perform an initial assessment of the change, to evaluate the scope of the change and the impact on existing deliverables (e.g. system GxP risk assessment, data classification assessment, user stories, etc.). This assessment should also address testing, SOP update, training and all aspects that have to be considered for the implementation and deployment of the change.

When a change implies the addition or updating of the backlog entries (user stories), the system GxP risk assessment and data classification assessment should be reviewed to ensure their validity and updated if required. The specific user story risk assessment for the new or updated ones has to be done.

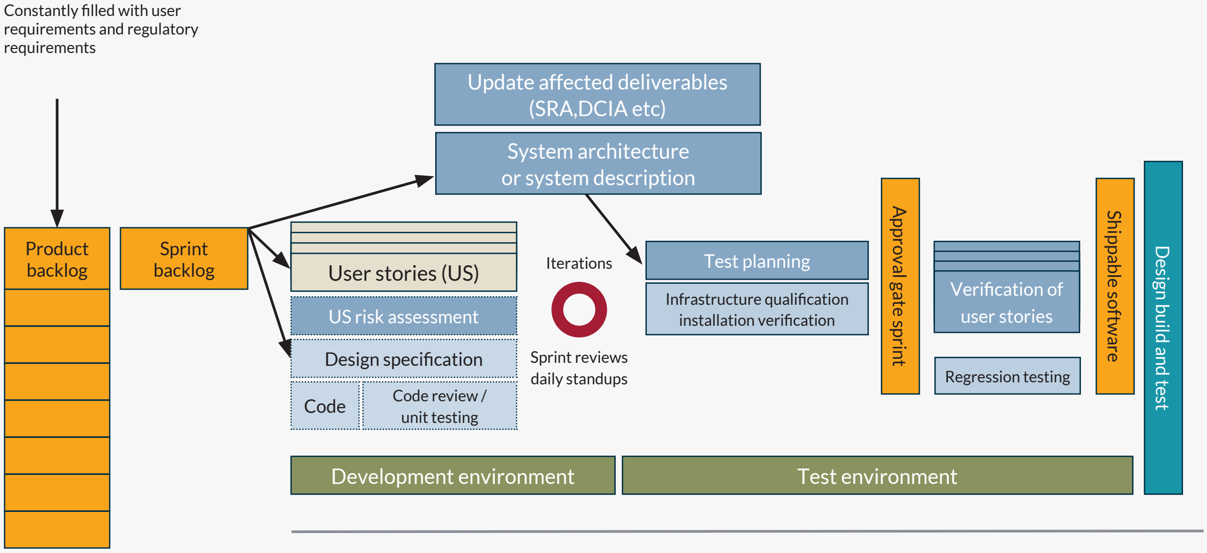
If the change does not modify the exiting backlog, the system GxP risk assessment and the data classification assessment are still valid.

Approval gate change planning

In order to pass this gate, the change ticket record has to be approved for design, build and test.

10.0 DEFINE, BUILD AND TEST STAGE

Diagram 7: Operation phase – design build and test


Constantly filled with user requirements and regulatory requirements
Product backlog
Sprint backlog
Update affected deliverables (SRA,DCIA etc)
System architecture or system description
User stories (US)
US risk assessment
Design specification
Code
Code review / unit testing
Iterations
Sprint reviews daily standups
Test planning
Infrastructure qualification installation verification
Approval gate sprint
Verification of user stories
Regression testing
Shippable software
Design build and test
Development environment
Test environment

Update affected deliverables (system GxP risk assessment, data classification assessment etc.)

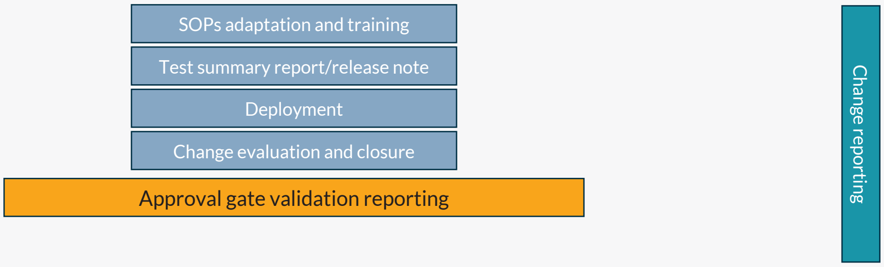
The affected deliverables, which have been identified in the change impact assessment (CIA) document, have to be updated.

Sprint backlog and further development to a shippable software package

Based on the change request, the affected features of the product backlog will be put into the sprint backlog. These features will be developed according to the process described in the design, build and test stage of the implementation phase. There is no difference to the approach described in that section.

11.0 CHANGE REPORTING STAGE

Diagram 8: Operation phase – change reporting


SOPs adaptation and training
Test summary report/release note
Deployment
Change evaluation and closure
Approval gate validation reporting
Change reporting

SOPs adaptation and training

The new and/or changed features have to be integrated into SOPs. Required training has to be conducted.

Test summary report/release note

The test summary report will summarize the testing outcome and any open defect will be assessed and classified. The release note will give an overview of the newly available features. When using an appropriate testing tool, the test report may be generated from the tool. This could also be possible for the release note.

Deployment

The deployment will ensure a seamless installation of the new software package on the productive environment and handover to the business users and operation team(s).

Change evaluation and closure

A final assessment has to be performed to control whether all activities that were planned in the initial CIA have been executed and all affected deliverables and records have been updated or created. Also, the effectiveness of the change should be evaluated.

Approval gate change reporting

In order to pass this gate, the change ticket record has to be approved for closure.

12.0 KEY CONSIDERATIONS

There should be a single comprehensive validation plan defining the strategy to be followed according to this guideline.

When an Agile approach is used for regulated environments, the user stories must be categorized based on their regulatory impact.

When a sprint is categorized as having a GxP impact the QA unit needs to be involved (reviewing any pending issue and approving the release to a production environment in a change control).

When Agile is used there are supporting tools to ensure that the methodology can be executed smoothly and in a true Agile way. (Agile cannot be applied without supporting and collaborative tools). The minimum supporting tools required are those to electronically manage the following records:

- change controls (change tickets) with e-approvals
- product backlog as a whole and user stories
- software releases and version control
- test cases (allowing traceability, referencing user stories), test runs/execution and defects.

These supporting tools must be validated and must have audit trail for the key records needed to show that the computer system is being operated in a state of control. Non-modifiable audit trails should be similar to the ones required by Part 11 (ER/ES), but the bare minimum should be:

- recording of entries for every modification to the above records (version control)
- compliant timestamp
- recording of the Individual performing the update.

From system documentation to electronic records

We have to consider that in Agile development environments we are changing documents and approvals by electronic records subject to version control and audit trails in validated tools.

When Agile is used for the implementation and maintenance of validated and regulated computer systems, the retention rules for the regulated records and associated metadata must be adhered to.

These supporting records are subject to the same retention rules as those managed through the traditional validation and development approaches, and the key ones directly supporting the validation of the system (e.g. user stories, test evidences, etc.) must be considered electronic records and must have a proper audit trail.

Therefore, when we apply an Agile development methodology supported by specific applications we have to ensure that:

- they are validated
- they are capable of keeping the regulated records for their specific retention period or able to archive them in a easily retrievable way for the applicable retention period.

Key regulated e-records subject to retention

The following records are to be considered electronic records and subject to retention rules when managed in electronic format:

- user stories, all versions
- change controls associated to each sprint
- software (code) releases, all versions
- test cases, all versions
- test execution and associated evidences (if any), last run defects
- test execution reviews and traceability (information) to user stories, last versions
- when applicable, change controls to the infrastructure components
- system description/architecture
- audit trails/logs associated to the above records.

13.0 CONCLUSIONS

The biopharmaceutical industry continues to be challenged to deliver medicines in a timely, cost-effective and safe way. The development and implementation of systems using an Agile approach supports this need. However, the Agile challenge has been to deliver system solutions in a timely, cost-effective way while delivering a validated solution.

This document provides guidance on how this can be achieved based on the collective knowledge and experience of the BioPhorum Operations Group IT (BPIT) member companies involved in the development of this paper. What has become clear is that the joint objectives of delivering a system solution in a quick and cost-effective way and a validated solution in a regulated environment are not mutually exclusive. Both can be achieved if the guidance provided in this paper is followed.

Appendix A – Agile Manifesto and key principles

http://agilemanifesto.org/principles.html

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development.
- The sponsors, developers and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity—the art of maximizing the amount of work not done—is essential.
- The best architectures, requirements and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Appendix B – Definitions

- Ceremony: a ritualistic or symbolic activity that is performed on well-defined occasions. Some people refer to the core Scrum activities of sprint planning, daily scrum, sprint review and sprint retrospective as ceremonies.
- Cross-functional team: a team composed of members with all the functional skills (such as user interface designers, developers, testers) and specialties necessary to complete work that requires more than a single discipline.
- Sprint review: an inspect-and-adapt activity that occurs after sprint execution where the Scrum team shows to all interested parties what was accomplished during the sprint. The sprint review gives everyone with input into the product development effort an opportunity to inspect what has been built so far and adapt what will be built next.
- INVEST: an acronym coined by Bill Wake for remembering a set of criteria used to evaluate the quality of user stories. The criteria are: Independent, Negotiable, Valuable, Estimable, Sized correctly (small) and Testable.
- Daily standup: synchronization, inspection and adaptive planning activity that a development team performs each day. This core practice in the Scrum approach is time boxed to no more than 15 minutes. Synonymous with daily scrum.
- Product backlog: a prioritized inventory of yet-to-be-worked-on product backlog items.
- Sprint backlog: a list of features, user stories or tasks that are pulled from the product backlog for consideration for completion during the upcoming sprint. Product backlog features and user stories are broken down into tasks to form the sprint backlog during the sprint planning meeting.
- Product vision: a brief statement of the desired future state that would be achieved through the project initiative. The product vision may be expressed in any number of ways, including financial performance, customer satisfaction, market share, functional capability, etc. The product vision is typically the responsibility of executive sponsorship and is articulated to the Agile development team by the business and by the product owner if the team is using Scrum.
- User story: a convenient format for expressing the desired business value for many types of product backlog items. User stories are crafted in a way that makes them understandable for both business people and technical people. They are structurally simple and typically expressed in a format such as "As a <User Role>, I want to achieve <goal>, so that I get <benefit>. They provide a great placeholder for conversation. Additionally, they can be written at various levels of granularity and are easy to progressively refine.
- Sprint retrospective: an inspect-and-adapt activity performed at the end of every sprint. The sprint retrospective is a continuous improvement opportunity for a Scrum team to review its process and to identify opportunities to improve it.
- Scrum Master: the coach, facilitator, impediment remover and servant leader of the Scrum team. Scrum Master is one of three roles on a Scrum team. The Scrum Master provides process leadership and helps the Scrum team and the rest of the organization develop their own high-performance, organization-specific Scrum approach.

Appendix C – Acronyms

Acronym/abbreviation	Definition
CSV	Computer Systems Validation
COTS	Commercial Off The Shelf
DCR	Data Classification Report
EDMS	Enterprise Document Management System
ERP	Enterprise Resource Planning
ER/ES	Electronic Record/Electronic Signature
GMP	Good Manufacturing Practices
GxP	Good x Practices, where x can be Laboratory, Clinical, Manufacturing
MES	Manufacturing Execution System
QA	Quality Assurance
SA	Solution Architecture
SaaS	Software as a Service
SOP	Standard Operating Procedure
SRA	System Risk Assessment

Appendix D – Things to consider from an inspector perspective

At the end of the process the expectation of the competent authorities (regulators) is that your records (paper, electronic or hybrid) will clearly demonstrate:

- what was supposed to be built
- what was actually built
- what testing was done
- what build or version of the system was the validation report issued against.

At some stage there will be a 'finalized specification document'. This is likely to have been a fluid document during the build and will have evolved, but at some point you need to say "that's it".

You need to be able to say what the final specification is, capture and record it.

Formal approval processes are required prior to the developers starting work. You need to be clear:

- what does and doesn't get included in each sprint
- informal meeting minutes are not acceptable
- what has and hasn't been included and why
- what aspects of functionality were covered in which sprint

A further formal approval process is required of the developers work:

- can you prove the tests were done?
- are formalized test scripts in place?
- approved test script and approval of completed test scripts, with the results.

A comprehensive tracking process is required to identify testing that fails.

Can you prove what was done?

- formal project 'deviation' system to record non compliances with the planned activities retained in a documented record—paper, electronic or hybrid. The record will demonstrate that the issues were investigated and either corrected, retested or passed
- no further retesting required (explain why not)

Permission to use
The contents of this report may be used unaltered as long as the copyright is acknowledged appropriately with correct source citation, as follows "Entity, Author(s), Editor, Title, Location: Year"

Disclaimer
This document represents a consensus view, and as such it does not represent fully the internal policies of the contributing companies.

Neither BPOG nor any of the contributing companies accept any liability to any person arising from their use of this document.

BioPhorum
Operations Group
Connect · Collaborate · Accelerate