**BDD - A Practicable Approach for Computerised System Validation?**

**Bachelor Thesis 2020**

Client: wega Informatik AG  
Author: Sabrina Leuenberger  
Lecturer: Stephan Jüngling  
City, Date: Basel, 23th of July 2020

|  |  |
| --- | --- |
| **BDD – A Practicable Approach for Computerised System Validation?** |  |
| **Author** |  |
| Sabrina Leuenberger Merlachfeld 54 3280 Murten +41 (0)78 935 19 99 saleuenberger@gmx.ch |  |
| **Lecturer** |  |
| Stephan Jüngling University of Applied Sciences and Arts  Northwestern Switzerland  stephan.juengling@fhnw.ch |  |
| **Client** |  |
| wega Informatik AG Mathias Fuchs und Evelyne Daniel Aeschengraben 20 4051 Basel +41 (0)61 270 87 87 info@wega-it.com www.wega-it.com |  |
| Basel, July 2020 |  |

Declaration of Authenticity

I the undersigned declare that I have prepared the present paper independently and without the use of sources other than those indicated in the reference list.

All statements and information contained herein are listed and indicated as quotations and / or paraphrases.

This Bachelor Thesis / Project Work / Student Research has not been published to date. It has thus not been made available to other interested parties or examination boards.

City, Date

Signature

Preface or Background of the Project or Acknowledgment

Acknowledgements:

* wega Informatik AG for hosting the project, access to Teams, Confluence and Jira as collaboration tools
* Evelyne Daniel and Mathias Fuchs from wega for all the support and valuable inputs, their time, ….
* Andreas Hosbach from Zühlke AG for all is support from the technical side, ……
* Stephan Jüngling for supervising this project from FHNW,……

Management Summary / Abstract

Table of Contents

[Declaration of Authenticity II](#_Toc38194517)

[Preface or Background of the Project or Acknowledgment III](#_Toc38194518)

[Management Summary / Abstract IV](#_Toc38194519)

[Table of Contents V](#_Toc38194520)

[1 Introduction 1](#_Toc38194521)

[1.1 Initial Situation 1](#_Toc38194522)

[1.2 Test Optimisation using Automated Testing 1](#_Toc38194523)

[1.3 BDD High Level Test Automation for OQs 2](#_Toc38194524)

[1.4 Tools 2](#_Toc38194525)

[1.5 Hypothesis and Research Questions 2](#_Toc38194526)

[1.6 Scope 3](#_Toc38194527)

[1.6.1 In Scope 3](#_Toc38194528)

[1.6.2 Out of Scope 3](#_Toc38194529)

[1.7 Approach 4](#_Toc38194530)

[2 Materials & Methods 6](#_Toc38194531)

[2.1 Analysis 6](#_Toc38194532)

[2.2 Prototyping 6](#_Toc38194533)

[2.2.1 Tools Included in the Evaluation 6](#_Toc38194534)

[2.2.2 Tools Used Outside the Evaluation Part 6](#_Toc38194535)

[2.2.3 Used tools 6](#_Toc38194536)

[2.2.4 Method 7](#_Toc38194537)

[2.3 Evaluations 8](#_Toc38194538)

[2.3.1 Tool-Evaluation 8](#_Toc38194539)

[2.3.2 Audit of the Prototype 8](#_Toc38194540)

[3 Computerised System Validation according to GAMP5 9](#_Toc38194541)

[3.1 GAMP5: An Overview 9](#_Toc38194542)

[3.1.1 What is GAMP5 9](#_Toc38194543)

[3.1.2 Key Concepts 9](#_Toc38194544)

[3.1.3 Life Cycle Approach 9](#_Toc38194545)

[3.1.4 Product Categories 9](#_Toc38194546)

[3.1.5 Project Phase for Category 5 Software 9](#_Toc38194547)

[3.1.6 Automated Testing 9](#_Toc38194548)

[3.2 OQ Process According to GAMP5 10](#_Toc38194549)

[3.2.1 Roles and Responsibilities 11](#_Toc38194550)

[3.2.2 The Main Process 12](#_Toc38194551)

[3.2.3 Incorporating the Quality Risk Assessment 13](#_Toc38194552)

[3.2.4 Specification- and Test Management 13](#_Toc38194553)

[3.2.5 Validation Activities needed in Respect to OQ Testing Tools 15](#_Toc38194554)

[3.2.6 Exemplary OQ Process 15](#_Toc38194555)

[4 Behaviour Driven Development 16](#_Toc38194556)

[4.1 BDD a suitable Software Engineering Approach for Highly Regulated Environments 16](#_Toc38194557)

[4.2 The Approach: An Overview 16](#_Toc38194558)

[4.3 Writing Executable Specifications with Gherkin 16](#_Toc38194559)

[4.4 Automation Tools 16](#_Toc38194560)

[5 OQs using BDD – A combined process 17](#_Toc38194561)

[5.1 Organisation, Roles, Responsibilities 17](#_Toc38194562)

[5.2 QA Process 17](#_Toc38194563)

[5.3 Test Automation Tool Validation for Cucumber/Gherkin, Scenarioo, Selenium and their interactions 17](#_Toc38194564)

[6 Prototyping 18](#_Toc38194565)

[6.1 Tool-Validation 18](#_Toc38194566)

[6.2 Application Design 18](#_Toc38194567)

[6.3 Environmental Set-Up 18](#_Toc38194568)

[6.3.1 General Set-Up 18](#_Toc38194569)

[6.3.2 OQ Test App 18](#_Toc38194570)

[6.3.3 JBA Frontend (JavaBusinessApplication) 19](#_Toc38194571)

[6.3.4 JBA Backend (JavaBusinessApplication) 19](#_Toc38194572)

[6.3.5 Database 19](#_Toc38194573)

[6.4 Specification/Formulation 19](#_Toc38194574)

[6.4.1 From User Stories to Feature Files 19](#_Toc38194575)

[6.4.2 Risk Assessment 19](#_Toc38194576)

[6.4.3 Compliance 19](#_Toc38194577)

[6.5 Test Automation 19](#_Toc38194578)

[6.5.1 Glue Files 19](#_Toc38194579)

[6.5.2 Test Reports 19](#_Toc38194580)

[6.6 QA Processes 20](#_Toc38194581)

[6.6.1 Specification and Test Management 20](#_Toc38194582)

[6.6.2 Traceability 20](#_Toc38194583)

[6.6.3 Approval 20](#_Toc38194584)

[6.7 Implications of adding new functionalities on the OQ Test-Suite 20](#_Toc38194585)

[7 Results of the Prototype Audit 21](#_Toc38194586)

[7.1 xxxx 21](#_Toc38194587)

[7.2 xxxx 21](#_Toc38194588)

[8 Learnings & Discussion 22](#_Toc38194589)

[8.1 xxxx 22](#_Toc38194590)

[8.2 xxxx 22](#_Toc38194591)

[9 Outlook 23](#_Toc38194592)

[9.1 General 23](#_Toc38194593)

[9.2 Specific Topics 23](#_Toc38194594)

[9.2.1 PQs 23](#_Toc38194595)

[9.2.2 Category 4 Software 23](#_Toc38194596)

[9.2.3 DevOps 23](#_Toc38194597)

[10 Formatting 25](#_Toc38194598)

[10.1 Quotes 25](#_Toc38194599)

[10.2 Enumerations 25](#_Toc38194600)

[10.3 Footnotes 25](#_Toc38194601)

[10.4 Figures 25](#_Toc38194602)

[10.5 Tables 26](#_Toc38194603)

[10.5.1 Tables as illustration 26](#_Toc38194604)

[10.5.2 Word tables 26](#_Toc38194605)

[10.6 Changing the font 27](#_Toc38194606)

[References 28](#_Toc38194607)

[List of Figures 30](#_Toc38194608)

[List of Tables 31](#_Toc38194609)

[List of Abbreviations 32](#_Toc38194610)

[Appendix 33](#_Toc38194611)

# Introduction

## Initial Situation

To ensure patient safety, software that directly or indirectly influences product quality must be validated according to Good Automated Manufacturing Practices (GAMP) (Wikipedia, 2018; International Society for Pharmaceutical Engineering ISPE, 2008, pp. 14, 15 and 27). This means that a formal and objective proof must be provided that the software is compliant and that its intended use is achieved (International Society for Pharmaceutical Engineering ISPE, 2008, p. 14; Johner, 2017). In support of this validation process, GAMP5 is a guide on how to achieve computerised system validation (CSV) (International Society for Pharmaceutical Engineering ISPE, 2008, p. 14).

According to GAMP5, software validation includes among others, the verification of user requirements (PQs) and of functional specifications (OQs) (International Society for Pharmaceutical Engineering ISPE, 2008, p. 38). Until today the wega CSV specialists team experienced that PQs and OQs are often performed manually, even though test tools like hp alm (Guru99, 2020a) are supporting testing documentations in regulated companies (Evelyne Daniel, personal communication, December 19, 2019 and April 1, 2020).

## Test Optimisation using Automated Testing

As described in several blogs, manual testing has the disadvantages that it is prone to mistakes and errors while performing the tests and that it is time consuming for human testers and therefore it generates high costs (Guru99b, 2020; Hoogenraad, 2017). This is especially true, when the same tests have to be performed several times (Guru99b, 2020). On the other hand, and in contrast to automated testing, manual testing allows to evaluate aspects like user friendliness and positive customer experience (Guru99b, 2020). Therefore this project will focus its investigations on test automation for OQs as testing on functional level does not include evaluations on usability aspects (Qualitest, n.d.).

Coming back to the situation in the pharmaceutical industry, Jae Burnett investigates in her paper ‘Practical Use of Automated Tool in Computer System Compliance’, the usage of test automation tools for validated pharmaceutical environments: She emphasises that life science companies should consider automated testing as an opportunity that could add significant value to the computer system compliance process (Burnett, 2009, p. 75).

She further suggests, that if a full test automation might not always be possible, as it might be difficult to integrate the formal approval and the control of test cases into the test automation system, hybrid approaches between automation and manual process could be considered (Burnett, 2009, p. 75).

In conclusion, OQs are interesting to investigate in respect of test automation, as they are so far mainly performed manually and have therefore a good potential of optimisation in respect of reliability and cost effectiveness. And in contrary to PQs there is no need to consider aspects of user friendliness and positive customer experience, for which automated testing is not suitable.

## BDD High Level Test Automation for OQs

OQs are considered as high level testing as they verify that the functional specifications are fulfilled (International Society for Pharmaceutical Engineering ISPE, 2008, p. 38).

Behaviour driven development (BDD), on the other side, is a software development approach that includes tools for high level test automation (Smart, 2015, p.25). One of such tool is Cucumber (Smart, 2015, p.25). As the test automation script is based on a formalised natural language, that is human and machine readable, it has the potential to be a powerful asset in the sense of a hybrid approach between automation and manual processes (Burnett, 2009, p. 75; Nagy & Rose, 2018, chapter 4.6).

In the same sense, Gáspár Nagy and Seb Rose state in their book ‘Discovery: Explore behaviour using examples’ that BDD is well suited for software development in regulated areas and refer thereby also to the U.S. Food and Drug Administration (FDA) (Nagy & Rose, 2018, chapter 4.6).

## Tools

* Cucumber/Gherkin which allow to automate the test cases using an automation script (gherkin feature file) (SmartBear Software, 2020).
* Selenium which simulates the user interaction with the web application and will be controlled by cucumber and the gherkin feature file (Selenium, n.d; Jain & Sawant, 2018).
* Scenarioo that is used to display test reports with screenshots (Scenarioo, n.d.).

## Hypothesis and Research Questions

This project will be led by following hypothesis: BDD with its activities from user stories to executable specifications (formulation) and automation is a practicable approach in respect of technical feasibility, taking into account Cucumber/Gherkin, Selenium and Scenarioo, and validation requirements according to GAMP5 for OQ test automation in highly regulated environments of the pharmaceutical industry.

To evaluate the above mentioned hypothesis, the project should find answers to following questions:

* Do the artefacts out of the BDD process satisfy the GAMP5 requirements in respect of OQs?
* Can automation tools like Cucumber/Gherkin, Selenium and Scenarioo (see chapter 2.2.1) be used together in validated environments?
* How can the test suite be adapted to the evolution of the application?
* How could be dealt with new versions of the automation tools in terms of validation?

## Scope

### In Scope

* Evaluation of a test automation for OQs based on BDD for a category 5 software (custom application) according to GAMP5.
* Proposal for a Validation Procedure for Cucumber/Gherkin, Scenarioo and Selenium and their interaction.
* Implementation of a prototype for the evaluation and illustration of a test automation for different functionalities. The test automation is based on the following tools: Cucumber, Selenium, Scenarioo.
* Are to be included: User Requirements, Specification/Test Management, Risk Management, Traceability, the validation process with regard to the OQs for a Category 5 product according to GAMP5.
* If possible, outlooks on related topics (PQs, category 4 software, DevOps), based on the findings obtained.

### Out of Scope

* Validation activities according to GAMP5 outside of OQs (e.g. process validation, IQs, PQs, design reviews, used infrastructure ...).
* BDD activities that are outside the chain from user stories to high-level test automation (e.g. the implementation technique TDD, Unit/Module Testing).
* Tool evaluations for implementation (e.g. Selenium vs. Cypress) are not part of this work.
* Risk evaluation regarding the implementation of the prototype: exemplary risks are considered, but without claiming that the risk evaluation was carried out correctly and completely from a practical point of view.
* Compliance of the prototype: if possible, an exemplary compliance requirement that is technically easy to implement will be considered. However, the prototype will not be compliant to regulations like FDA 21 CFR part 11 or EU GMP Annex 11.

## Approach

The starting point for the BDD part is the book "Discovery - Explore behaviour using examples" by Gaspar Nagy and Seb Rose (ISBN 978-1983591259). In chapter 4.6 the authors state that BDD leads to improved efficiency in software development for regulated environments.

The CSV part of the project is based on the widely-used CSV standard GAMP5.

The BDD approach and the GAMP5 methodology are analysed to develop a combined process. This process is then presented in an appropriate form (e.g. BPMN) so that it can be evaluated by a wega CSV specialist. The process will be regularly reviewed by a wega CSV specialist while being developed to assure its conformity with GAMP5. Based on the developed process a prototype will be implemented as proof of concept. Few exemplary user requirements are defined as a basis for its implementation. As the final input regarding the suitability of the developed model, the prototype is audited by a wega CSV specialist. This then forms the basis for the ‘Learnings & Discussion’ and the ‘Outlook’.

In summary following activities are planned in two different streams as shown in the figures 1 & 2:

* Development of a combined process between BDD and GAMP5 in respect to OQs
* Based on point 1 to implementation of a prototype
* Audit of the prototype by wega
* Analysis of the audit results including learnings, a discussion and an outlook
* Analysis of what is needed to be done in order to use Cucumber/Gherkin, Scenarioo and Selenium in a validated environment including following aspects
  + single tools
  + combination of the tools
  + updates of the tools

Figure 1 Process to investigate OQ test automation

Figure 2 Analysis of the usability of automation tools for regulated environments

# Materials & Methods

## Analysis

Literature search: Semantic scholar, google, google scholar, swissuniversities databases

BPMN: <http://www.bpmn.org/>

Draw io desktop app from draw.io

C4 model for software architecture (<https://c4model.com/>).

## Prototyping

### Tools Included in the Evaluation

* Cucumber/Gherkin which allow to automate the test cases using an automation script (gherkin feature file) (SmartBear Software, 2020).
* Selenium which simulates the user interaction with the web application and will be controlled by cucumber and the gherkin feature file (Selenium, n.d; Jain & Sawant, 2018).
* Scenarioo that is used to display test reports with screenshots (Scenarioo, n.d.).

### Tools Used Outside the Evaluation Part

The use of the following tools is anticipated but might be subject to changes. These tools will support the work but are not relevant for the evaluation itself.

* Front end: vue CLI 🡪 <https://cli.vuejs.org/>
* Back end: Spring 🡪 <https://spring.io/>
* Database: PostgreSQL 🡪 <https://www.postgresql.org/>
* Others:  
  Maven 🡪 [https://maven.apache.org/](https://maven.apache.org/guides/introduction/introduction-to-repositories.html)   
  GitHub 🡪 <https://github.com/>

### Used tools

OQ TEST APP

* AdoptOpenJDK 14 with HotSpot as JVM (200406 - <https://adoptopenjdk.net/index.html?variant=openjdk14&jvmVariant=hotspot>) was chosen as it is an open-source version of the Java Standard Edition platform (200406 - <https://en.wikipedia.org/wiki/OpenJDK>). There are different Open JDK distributors on the market (200406 - <https://en.wikipedia.org/wiki/OpenJDK>). AdoptOpenJDK was chosen as it is recommended by stackoverflow when no specific environmental or license requirement are needed and the most standard DK build would therefore be appropriate (200406 - <https://stackoverflow.com/questions/52431764/difference-between-openjdk-and-adoptopenjdk>) .
* IntelliJ IDEA 2019.2.4 Community Edition was used as development environment: 200406 - <https://www.jetbrains.com/idea/>
* Maven version 3.6.1 was used to allow dependency management: for description see also the oq-test-app POM on <https://github.com/sableu/BDD4OQ>.......
* JUnit Jupiter is used as testing framework on the Java Virtual Machine (JVM). It is used in the version as described in the oq-test-app POM on <https://github.com/sableu/BDD4OQ>.......
* Cucumber/Gherkin and Selenium (200415 - <https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java>) were imported via the maven pom file and were used in the versions as described in the oq-test-app POM on <https://github.com/sableu/BDD4OQ>.......
* The Cucumber-Scenarioo-plugin version 0.1.0 was downloaded from (200406 (<https://github.com/andreashosbach/cucumber-reporter>) , installed in the local maven repository by creating a maven build as described (200408 - <https://maven.apache.org/guides/getting-started/index.html>) and integrated as maven dependency as described in the oq-test-app POM on <https://github.com/sableu/BDD4OQ>.......
* Scenarioo release 5.0.2 was setup and used in the standalone application version as described in (200408 - <http://scenarioo.org/docs/master/tutorial/Scenarioo-Viewer-Web-Application-Setup.html#setup-1---running-as-standalone-application>)
* Chrome Browser version 80.0
* Chrome Driver version 80.0 downloaded from (200406 - <https://chromedriver.chromium.org/downloads>) to allow automated testing by Selenium (200406 - <https://chromedriver.chromium.org/>)

### Method

Incremental Software Engineering Approach in order to evaluate the question, what implication the extension of the application with new functionalities on the test suite (see chapter 6.6).

## Evaluations

### Tool-Evaluation

Tool Evaluation for Cucumber/Gherkin, Scenarioo and Selenium: An analysis will be performed on what is needed in order to be able to use these tools in a validated environment.

### Audit of the Prototype

The audit will be done by wega. The goal of the audit is to evaluate the documents and the prototype itself that were created within the scope of this project, in respect to the GAMP5 requirements for OQs of a category 5 software. The audit report is attached to this document.

# Computerised System Validation according to GAMP5

## GAMP5: An Overview

### What is GAMP5

### Key Concepts

### Life Cycle Approach

### Product Categories

### Project Phase for Category 5 Software

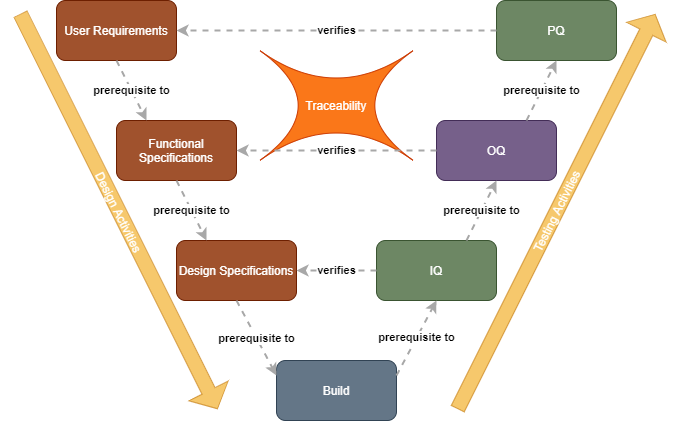
### Automated Testing

(International Society for Pharmaceutical Engineering ISPE, 2008, pp. 207)

## Verification for Custom Applications According to GAMP5

Verification activities for the implementation of a custom application are about demonstrating that the software is compliant and fit for intended use by confirming that specifications have been fulfilled (GAMP5 pp. 31 and 37). GAMP5 foresees, that after the application has been built, an installation qualification (IQ), an operational qualification (OQ) and a performance qualification (PQ) are achieved by testing activities (GAMP5 p 38). In this sense a test strategy, also called testplan, should define and document among others, how IQ, OQ and PQ should be applied for the specific software, based on company procedures that were established to define the general framework for testing (GAMP5 pp. 196 and 201).

The figure below shows in an abstract form how specifications are related to the different testing activities (GAMP5, p. 36; 200417 - <https://www.softwaretestinghelp.com/iq-oq-pq-software-validation/> ; 200419 - <https://www.blazesystems.com/blaze-iq-oq-pq.html> ; 200419 - <https://www.ciprecision.com/validation-services/>):



In view of OQ automation, tt has to be noted, that the OQ is performed on the fully built and installed software. In order to respect this process, that OQ is performed after installation, it will not be possible to use automation tools that perform testings during the build process, as for example unit testing using Junit (look for reference). Therefore, an OQ Test App will be needed, that uses the interfaces of the deployed application.

It must also be pointed out that traceability between the individual specifications (from user requirements to functional specifications and design specifications) as well as to the corresponding tests must be ensured throughout the entire process (GAMP5, pp. 134-137).

## OQ Process According to GAMP5

GAMP5 defines OQs the following way: “Operational Qualification (OQ)” [...is a...] “documented verification that a system operates according to written and pre-approved specifications throughout specified operating ranges (GAMP5 p. 38). Whereas GAMP5 foresees following verification activity: “Testing or other verification of the system against specifications to demonstrate correct operation of functionality that support the specific business process throughout all specifies operating ranges” (GAMP5 p. 38). In the GAMP5 appendix D5 p. 212 for custom applications, this is explicitly linked to functional testing, as it states that these testings should focus on functionality that supports the specific business process based on risk and supplier assessment which exactly corresponds to the wording used to describe OQs as seen before.

### The Main Process

Writing OQs starts in parallel with the development of the functional specifications by the supplier (e.g. the IT unit of the company (🡪 check for reference), by elaborating the corresponding test specifications (GAMP5 p199 in combination with p175). GAMP5 does not mention which role is foreseen for this. The test specifications describe the overall purpose and a description of a set of test scripts (GAMP5 p198). For example, it defines which resources are needed, including tools for automated testing, the version of software under test, the test scripts to be carried out, methods, prerequisites, required reviews and approvals, etc. (GAMP5 p199). In addition, GAMP5 requires some metadata about the test specification document (GAMP5 p199).

Based on the test specification, the test analyst is responsible for developing test scripts that describe the tests to be performed in such a way that the testers can execute them consistently (GAMP5 p199). Next to some metadata, test scripts consist among others, of a title, a test description, the test objective, prerequisites, test steps, acceptance criteria and instructions about data to be recorded (GAMP5 p199 f.).

GAMP5 foresees, that the test specifications and the test scripts can be recorded in a single document (GAMP5 p 200). From this statement it can be deduced, inter alia, that the test analyst is not only responsible for the test script elaboration (GAMP5 p 196), but also for writing the test specifications.

As already mentioned, the test scripts are executed by the testers (GAMP5 p 196). As a result of their testing, they have to deliver records that are reviewable. The records contain the result of the single tests (passed/failed) including needed descriptions and supporting documentations as defined in the test scripts for example screen shots (GAMP5 p 200).

Subsequent to the execution of the tests, the test results will be reviewed by the test reviewer, which should not be the same person as the tester, and the results recorded in the test report (GAMP5 p. 196 and p. 200). A test report includes again some metadata like who executed and who reviewed the testings, and information about the effected testings like a summary of the test results, a summary of test failures and conclusions (GAMP5 p 200).

### Incorporating the Quality Risk Assessment

The above described OQ process, is, according to GAMP 5 accompanied by supporting processes (GAMP5 p32). One of them is risk management (GAMP5 p32), which in fact is not only a supporting process, but which is considered as one of the five key concepts of GAMP5 (GAMP5 p.20). The goal of this concept is to focus validation efforts on critical points of the computerised system (GAMP5 p.20). The quality risk management process according to GAMP5 includes in a first step the identification of functions with impact on patient safety, product quality and data integrity (GAMP5 p.50). Responsible for this step is a team of subject matter experts and key users (p106), or if regulatory compliance is concerned, it is the quality unit (p106). For the next step, the team perform a functional risk assessment and identify controls, based on the advice of the supplier, to eliminate or at least mitigate the risk to an acceptable level (p48 in combination with 50 and 106). Appropriate controls, i.e. quality critical requirements (p. 164), need then to be implemented GAMP p.50, e.g. by the supplier for additional software functionalities (p19) and verified (GAMP p.50) e.g. in the OQ process (p.38).

As the goal of OQ is the documented verification of software functionalities (GAMP p.38 in combination with p212), it is important to take the identified risks, their risk level and the defined control measures in respect to software functionalities into account, as the specific level of test efforts should than be determined according to the risk level and the system impact (GAMP p.50) and the controls themselves might be subject to the OQs (GAMP5, p38). The risk assessment documentation needs finally be approved by the system owner and/or the quality unit (p106)

### Specification- and Test Management

Next to the quality risk assessment, there are other supporting processes to be considered in respect to OQs for a custom application (p32). They include:

* Change management process: “Change management procedures also should be established. The point at which change management is introduced should be defined. Appropriate change processes should be applied to both project and operational phases.” (p.32) While performing OQ no change of the software is expected, as it is done on the version for which in a previous step the IQ were performed and approved (p.209). Therefore, the change management process will not be further considered.
* Configuration management: “Appropriate configuration management processes should be established such that a computerized system and all its constituent components can be identified and defined at any point” (p.32). Bringing this back to the level of the OQ process, it has to be clearly stated on which version of the software the OQ is performed, i.e. the same version as the preceding IQs (p.209).
* Traceability is the process to ensure that requirements are covered and traced to the corresponding functional specifications and design components, which then must be linked further to the appropriate verification (p.33; p.134ff). This means for the OQ process, that each OQ test script needs to be traced back to the underlying functional specification, which on its turn needs to have a link to the requirements from which it is derived. In GAMP5 three types of requirements with impact on OQs via the means of traceability could be identified: User requirements (e.g. p.163), regulatory requirements (e.g. p.163) and quality critical requirements (e.g. p.164). The responsibility for the requirements lies within the regulated company p.165. The Subject Matter Experts (SME) together with the user and technical communities are responsible to develop and document appropriate requirements (p.165).
* Document management process: “Management of documentation includes preparation, review, approval, issue, change, withdrawal, and storage”. (p. 33). The process described by GAMP5 (p. 153ff.) can be adapted in order to fit to the complexity of the project (p. 153). In respect of the OQ process following points were identified to be the most important ones:
* The author is normally responsible for the document prior to its review and the document, which normally should be subject to version control, is in the status ‘draft’ (p154).
* The draft is then reviewed ideally by an independent SME and the subsequent actions should be resolved prior to approval and issue (p.154).
* The approval of a documents consists of a signature, a comment about the approval reason and a date (p. 154). The document index and - history should be updated and the new status set, i.e. from ‘draft’ to ‘approved’ (p.154). According to the role description of p. 196 of GAMP5, this could be the user in respect to the test specifications and the test scripts, as she/he is responsible for the system and the quality unit in respect to the test report.
* The approved document is being issued by updating the document index (p. 154). This could mean in respect to the OQ process, that the approved test scipts are handed over to the tester.
* Document changes needs to by controlled by, e.g., updating the document index and -history and by setting the status back to ‘draft’, thereby and in consequence repeating the document approval process as described before (p. 155). In respect to the OQ, it might be a result of the test report, that there are test script errors and therefore the concerned test scripts will be reset to ‘draft’ by the test report approver.
* Document withdrawal can be done by updating the document index, -history and status and information of any controlled copy holders (p.155)
* In respect of document records and storage GAMP5 states that they should be stored in a safe and secure way according to a defined process (p.155).

### Validation Activities needed in Respect to OQ Testing Tools

Commercial or established testing tools are normally considered to be GAMP Category 1 🡪 compare appendix M4 but an assessment needs to be done according to the appendix M4. It could also be imaginable to be of Category 5 or anything in between, depending on the used tools.

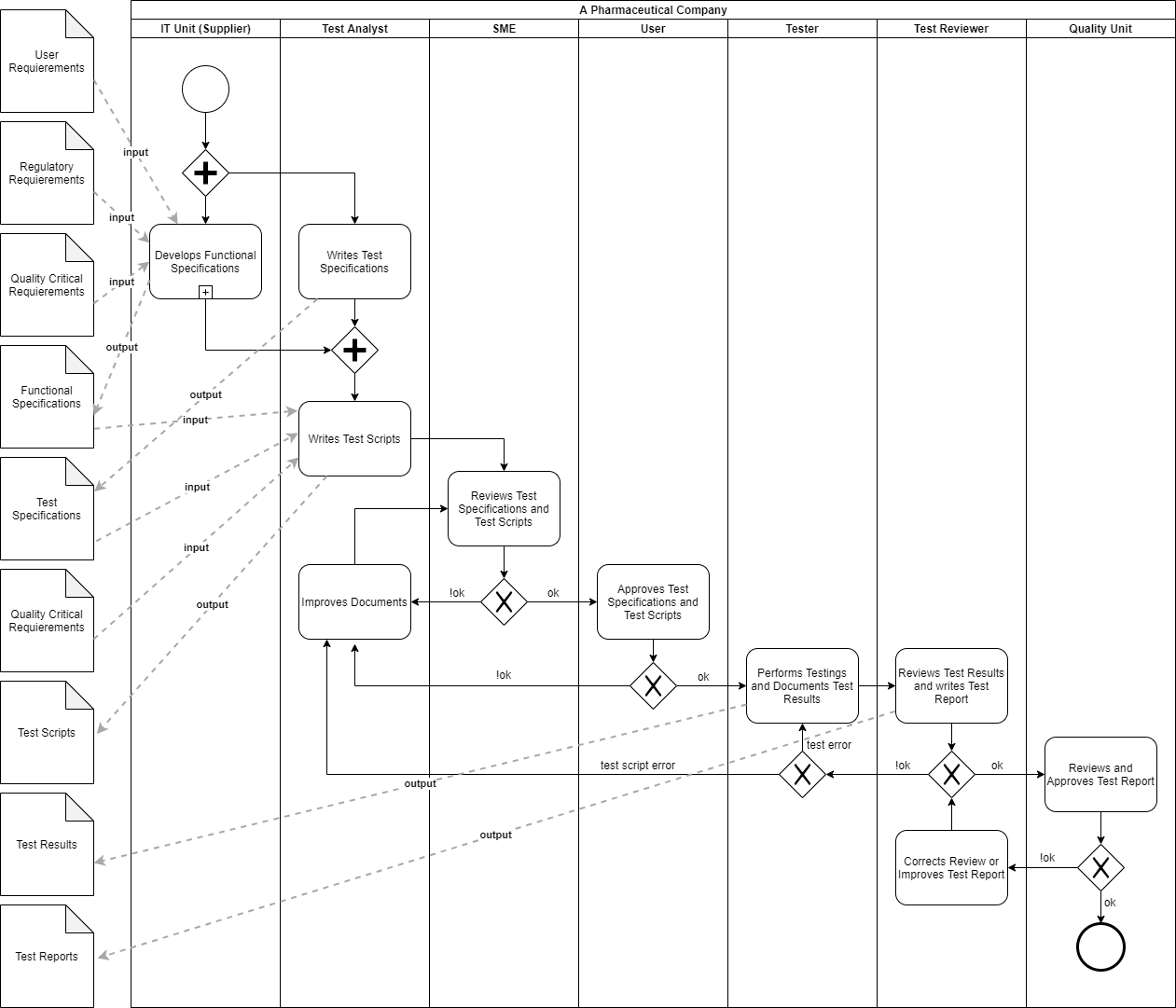
Analysis of the combination

Riskanalysis: ..........--> damit beginnen

### Exemplary OQ Process

Based on the descriptions on the earlier chapters, an exemplary process was developed, which highlights the most important tasks, roles and documents in respect to the OQ process. The further away one of these elements lies from the core OQ process, the less detailed the illustration falls out in order to keep it as clear and focused as possible.

This process is to be built upon in the further course of the project.



# Behaviour Driven Development

Literature:

* The BDD Books: Discovery – Explore behaviour using examples   
  Gáspár Nagy and Seb Rose   
  ISBN: 978-1983591259
* BDD in Action – Behavior-Driven Development for the whole software lifecycle  
  John Ferguson Smart  
  ISBN: 9781617291654
* Writing Great Specifications – Using Specification by Example and Gherkin  
  Kamil Nicieja  
  ISBN: 9781617294105

## BDD a suitable Software Engineering Approach for Highly Regulated Environments

## The Approach: An Overview

## Writing Executable Specifications with Gherkin

* Basierend auf dem Buch: Writing Great Specifications: Using specification by example and Gherkin von Kamil Nicieja

## Automation Tools

BDD Automation Tool:   
- Cucumber/Gherkin: <https://cucumber.io/>

Further tools to complement Cucumber:  
- Scenarioo: <http://scenarioo.org/>  
- Selenium: <https://www.selenium.dev/>

# OQs using BDD

## The Combined Process

It was tried to define an exemplary process independent of the software development process (agile, waterfall, etc.). It could be all defined at the beginning of the project like in a waterfall or one could go through this process in several itarations, coming closer to an agile approach.

Formulation Team: Should consist on persons that do well know the practicle side of the Requirements (User, Regulatory and Quality Critical requirement, a tester, but not the persons having the tester role, as they should be independent, and one or several representatioves of the IT Unit).

## Test Automation Tool Validation for Cucumber/Gherkin, Scenarioo, Selenium and their interactions

## Conclusions

In principle it is possible to define an OQ process according to GAMP5. It does not need to be exactly the way as was shown before, but with this process we have a poof of concept, that using BDD could be done in a way that is compatible with GAMP5

Viel vernetzter, prozess der functional specifications entfällt, traceability sehr gut garantiert, aufwertung des Testers der auch eine Qualitätssicherungsfunktion übernimmt, die manuelle ausführung des tests (stupide Arbeit entfällt)

Ob Prozes weniger Aufwand bringt ist fraglich, aber stupide Arbeit entfällt und wird statt dessen hoffentlich in bessere Kontrolle und Qualität investiert 🡪 zum Beispiel überienstimmng der Feature Files und der Glue Files 🡪 Quality check.

Es lohnt sich ein prototype zu bauen um ein besseres Gefühl zu erhalten, auch wenn der Prototyp nur bedingt nach diesem PRozess entwickelt werden kann, da es kein Team hat.

# Prototyping

## Tool-Validation

Implication of updates of the used tools on validation

Category 1 (Selenium, Cucumber Gherkin nach GAMP5)

Category 3 (Scenarioo)

Category 5 (Cucumber Reporter Plugin)

Versionskontrolle ist wichtig 🡪 controlled by Maven, das hier über die Dependencies erlaubt es zu kontrollieren, deshalb ist Maven wichtig im Environmental Set-up

## Application Design

🡪 siehe dazu das draw.io diagram C4, level1

## Environmental Set-Up

### General Set-Up

For the development of the prototype following basic tools were used for all parts of the prototype:

Developement Environment

Maven

JUnit

Java14

### OQ Test App

Code is based on Code written by Andreas Hosbach (200406 -<https://github.com/andreashosbach/cucumber-reporter> )

### JBA Frontend (JavaBusinessApplication)

### JBA Backend (JavaBusinessApplication)

### Database

## Specification/Formulation

### From User Stories to Feature Files

### Risk Assessment

### Compliance

## Test Automation

### Glue Files

### Test Reports

## QA Processes

### Specification and Test Management

### Traceability

### Approval

## Implications of adding new functionalities on the OQ Test-Suite

# Results of the Prototype Audit

## xxxx

## xxxx

# Learnings & Discussion

## xxxx

## xxxx

# Outlook

## General

Selenium Testing Vor- Nachteile: Aufwändig wenn sich Oberfläche ändert, OQs können einfach auf verschiedenen Browsern ausgeführt werden 🡪 Testen der Applikation auf verschiedenen Browsern.

Digitization: Prozess möglichst klassisch gehalten, so wie er jetzt, manuell schon ist. Test automation passt sich diesem Prozess an. Denkbar wäre, OQs schon im Build-Prozess (inside Testing vs. Outside testing) durchzuführen. Würde aber bedeuten, dass die Reihenfolge IQ 🡪 OQ nicht mehr stimmen würde und dass die Qualificationen der Auditoren und der QA Menschen sich verschieben müssten 🡪 Müssten Code lesen und beurteilen können (vgl. Kleines BDD Buch)

## Specific Topics

### PQs

### Category 4 Software

### DevOps

# Formatting

## Quotes

Enable „Replace "Straight quotes" with “smart quotes”” in the AutoFormat section of the AutoCorrect options. Default language is „English (U.K.)“.

Variant: «…»

## Enumerations

Please use the „List Bullet“ template.

* Item 1
* Item 2

## Footnotes

Example for text containing a footnote.[[1]](#footnote-1)

## Figures

Please use the following option to insert graphical illustrations (e.g. Power Point charts):

Home Tab 🡪 Paste 🡪 Paste Special… 🡪 Picture (Enhanced Metafile).

Use the template „Figure“ for the formatting of illustrations. After inserting the illustration it must be anchored to the text. For this purpose, place the cursor in the target area of the figure (a paragraph which is formatted as „Figure“), do a right click on the figure and follow the instructions below:

Size and position 🡪 Text Wrapping 🡪 In Line with Text

The size of the illustration can also be changed in „Size and position“.

To **name** your illustration, place the cursor below your figure and choose „Insert Caption“ from tab „References“.

Caption: Figure number, colon, space, caption.

Please compare the following example: Figure 1



Figure 3: Example illustration

## Tables

### Tables as illustration

For tables which are imported as **illustration** or **object** e.g. from excel, use the same procedure like for figures (2.4) except that the illustration is formatted as “Figure Table” instead of “Figure”

For the caption use the same label “Table” as for word tables and format it as “Caption Table”.

Table caption: Below the table, like in our example: Table 1



Table 1: Example for a table imported as illustration

### Word tables

Choose our template „Table heading“ for title row and „Table text“ for table content. Please follow our instructions to caption your table:

Tab “References” 🡪 “Insert caption” (Position: Below selected item)

Reformat accordingly to „Caption Table“ (Styles).

Caption: Table number, colon, space, caption.

|  |  |
| --- | --- |
| Bezeichnung der  Grössenklasse | Anzahl Beschäftigte  (Vollzeitäquivalent) |
| Kleinunternehmen | 10 bis unter 50 |
| Mittlere Unternehmen 1 | 50 bis unter 100 |
| Mittlere Unternehmen 2 | 100 bis unter 250 |

Table 2: Example for a table created by Word

## Changing the font

The font for the whole document can be changed by the “normal” template (is not recommended). All other styles which are used in this document are linked to this template.

To change the font anyway:

Tab „Home“ 🡪 „Styles“ 🡪 Choose template „Normal“ in the dropdown-menu (right click) 🡪 „Modify“

References

Burnett, J. (2009). Practical Use of Automated Tools in Computer System Compliance. Journal of Validation Technology, 15(4), 73–76. Retrieved from https://www.ivtnetwork.com/journal-validation-technology/journal-of-validation-technology-3004

GitHub, Inc. (2020). Build for developers. Retrieved March 27, 2020, from https://github.com/

Guru99. (2020a, March 19). Introduction to HP ALM(Quality Center). Retrieved April 3, 2020, from https://www.guru99.com/hp-alm-introduction.html

Guru99. (2020b, March 23). Automation Testing Vs. Manual Testing: What’s the Difference? Retrieved April 3, 2020, from https://www.guru99.com/difference-automated-vs-manual-testing.html

Hoogenraad, W. (2017, October 11). Was ist manuelles Testen? Retrieved April 3, 2020, from https://de.itpedia.nl/2017/10/11/wat-is-handmatig-testen/

International Society for Pharmaceutical Engineering ISPE. (2008). GAMP 5 - A Risk-Based Approach to Compliant GxP Computerized Systems. North Bethesda, USA: ISPE.

Jain, N., & Sawant, T. (2018, February 5). Setup for Selenium with Cucumber Using Maven. Retrieved April 3, 2020, from https://www.axelerant.com/resources/team-blog/setup-for-selenium-with-cucumber-using-maven

Johner, C. (2017, April 5). Computer System Validation CSV. Retrieved March 29, 2020, from https://www.johner-institut.de/blog/regulatory-affairs/computer-system-validation-csv/

Nagy, G., & Rose, S. (2018). Discovery - Explore behaviour using examples. Victoria, Canada: https://leanpub.com/.

Nicieja, K. (2018). Writing Great Specifications. Shelter Island, NY, USA: Manning Publications.

Object Management Group. (1997 to present). Business Process Model and Notation. Retrieved March 27, 2020, from http://www.bpmn.org/

Qualitest. (n.d.). Functional Testing vs. Usability Testing. Retrieved April 3, 2020, from https://www.qualitestgroup.com/white-papers/functional-testing-vs-usability-testing/

scenarioo. (n.d.). Retrieved March 27, 2020, from http://scenarioo.org/

Selenium. (n.d.). Selenium automates browsers. That’s it! Retrieved March 27, 2020, from https://www.selenium.dev/

Smart, J. F. (2015). BDD in Action. Shelter Island, NY, USA: Manning Publications.

SmartBear Software. (2020). Tools & techniques that elevate teams to greatness. Retrieved March 27, 2020, from https://cucumber.io/

The Apache Software Foundation. (2002 to present). Maven. Retrieved March 27, 2020, from https://maven.apache.org/

The PostgreSQL Global Development Group. (1996 to present). PostgreSQL: The world’s most advanced open source database. Retrieved March 27, 2020, from https://www.postgresql.org/

VMware. (2020). Spring makes Java... Retrieved March 27, 2020, from https://spring.io/

Wikipedia. (2018, June 26). GxP. Retrieved March 29, 2020, from https://de.wikipedia.org/w/index.php?title=GxP&oldid=178646512

You, E. (2018 to present). Vue CLI. Retrieved March 27, 2020, from https://cli.vuejs.org/

List of Figures

[Figure 1 Process to investigate OQ test automation 4](#_Toc36826148)

[Figure 2 Analysis of the usability of automation tools for regulated environments 5](#_Toc36826149)

[Figure 3: Example illustration 19](#_Toc36826150)

Annotation: A convenient way to create default list of figures and list of tables in the text is to follow the sequence:

Tab „References“ 🡪 „Captions“ 🡪 „Insert Table of Figures“

Now, the caption is formatted automatically according to our template „caption“.

For the figures the template „Figure“ and for tables the templates „Table heading“ and „Table text“ should be used.

Further information can be found in chapters: 2.4 and 2.5

List of Tables

[Table 1: Example for a table imported as illustration 3](#_Toc416160904)

[Table 2: Example for a table created by Word 3](#_Toc416160905)

List of Abbreviations

|  |  |
| --- | --- |
| Abbreviation | Description |
| BDD | Behaviour Driven Development: A software development approach developed by Dan North covering the whole software lifecycle |
| CSV |  |
| GAMP |  |
| GAMP5 |  |
| FDA | U.S. Food and Drug Administration ….. |
| GxP |  |
| JVM | Java Virtual Machine |
| OQ | Operational Qualification: …. |
| PQ | Performance Qualification …. |
| SME | Subject Matter Expert |
|  |  |
|  |  |
|  |  |

Appendix

References and appendix are formatted as „Heading 1 without numbering.“ Thereby this entry is listed in the table of contents (but without numbering). To structure your appendices use CAPITAL LETTERS (Appendix A, Appendix B etc.).

1. Example Footnote. [↑](#footnote-ref-1)