Method æ; the agile software development method tailored for the pharmaceutical industry

A. Hajou, R. S. Batenburg, and S. Jansen, *Utrecht University, the Netherlands*

Abstract— **Agile software development methods have a high adoption rate due to their proven business benefits. Its success has not yet been identified in the pharmaceutical industry, due to the industry's incompatibility with generic agile methods. Comprehensive research has been done to engineer a tailored agile method driven by the needs of the industry. 14 interviews were conducted with domain experts to elaborate on problems they encountered during software development projects in the pharmaceutical industry. A total of 137 method requirements were extracted from the interviews to engineer an agile method tailored for the pharmaceutical industry. Method æ has planned and iterative phases, enforces risk oriented decision making and reduces the number of documents to an absolute minimum. Method evaluation proves that Method æ is inter alia compliant to the EU GMP Annex 11 regulation and can therefore serve as a substitute for software development projects in the pharmaceutical industry.**

Index Terms—**Agile, Software Development, Pharmaceutical Industry, Regulatory Compliance**

I. Introduction

Agile methods are increasingly popular in the area of software development. Since the introduction of the Agile Manifesto in 2001 [1] agile has received a steady increase of interest in the area of research as well in its application in the industry of software development. Dingsøyr, Nerur, Balijepally and Moe emphasize on the industry driven developments of methods that can be considered to be called 'agile': "*A host of methods, adhering to varying degrees to the tenets of the manifesto, appeared on the landscape. These include eXtreme programming (XP), scrum, lean software development, feature-driven development (FDD), and crystal methodologies, to name but a few*" [2]. These methods have proven to have a positive effect on the time-to-market, the ability to adjust to changing requirements and the alignment between business and IT.

However, the methods have a high adoption rate across many types of industries and project team sizes, but due fail to fully penetrate the pharmaceutical industry remaining it having to cope with high overhead IT projects [3], [4]. The lack of adoption is the strict governance that is applied in this sector. "*The pharmaceuticals industry is characterised as being an innovative but highly regulated sector*" [5]. The innovation in medicinal products is large, while development of peripheral activities (e.g. software development projects) is very much the topic of negligence. The pharmaceutical industry is characterised as a 'regulated environment', which often uses the V-Model to conduct IT projects [6]. The V-Model is considered to be a 'Waterfall' method, which "*emphasizes a structured progression between defined phases*" [7]. These models (i.e. V-Model or waterfall method) are characterised as highly governed but rigorous, "*heavy weight*", slow methodologies. In an era where software becomes more of an organisational primary necessity, the software development projects require a shift to a more flexible (i.e. agile) approach.

Unfortunately, "*Agile methods and regulated environments are often seen as fundamentally incompatible*" [3]. The pharmaceutical industry considers GAMP5 (i.e. Good Automated Manufacturing Practice, version 5) as its main industry standard for providing guidance in IT projects, as will be elaborated upon later. During the research, a total of 42 different documents have been distinguished in the guide, whereas agile emphasizes "*working software over comprehensive documentation*" [1]. While GAMP5 emphasizes that it "*is neither mandatory, nor prescriptive, but aims at enabling innovation in a compliant and cost effective manner*" [8], it, and together with GAMP5 more industry related IT project guides, is in its fundaments different from the agile philosophy [9], [6].

But what happens if we shift the focus from GAMP5 to regulation and legislation? As part of an elaborated research [10], an agile method has been engineered based on the needs of the industry (i.e. represented by industry experts in various IT and validation related domains) and the obligations of the EU directive; EudraLex, Volume 4, GMP Annex 11: Computerised Systems [11].

II. Research Approach

In our previous research [12] a total of 3 holistic problem categories (PC) have been identified that describe the difficulties of software development methods (e.g. agile) in the context of the pharmaceutical industry. These PCs have been denoted as 1; "*the regulatory complexity of software development in the pharmaceutical industry*", 2; the *"differences between the agile and the highly documentative approaches",* and 3; the "*lack of attempts to be agile in the pharmaceutical industry"* [12]. Especially PC3, which

The work described was sponsored by Magna Medical B.V. in Nijmegen, the Netherlands. We are grateful for the generous input and constructive guidance.

A. Hajou is with the Department of Information and Computer Science at Utrecht University, the Netherlands (e-mail: a.hajou@students.uu.nl).

dr. R. S. Batenburg is an assistant professor at the Department of Information and Computer Science at Utrecht University, the Netherlands (e-mail: r.s.batenburg@uu.nl).

dr. S. Jansen is an assistant professor at the Department of Information and Computer Science at Utrecht University, the Netherlands (e-mail: s.jansen@uu.nl).

primarily is describing the lack of conducted scientific research and the absence of attempts to perform software development projects using agile methodologies required practical substantiation. Therefore these 3 PCs have been split up in 14 isolated problem subjects (i.e. abbreviated to PS, extracted from the PC descriptions) to be used for further elaboration during expert interviews.

PC1 consisted of the PSs;

- "*Multi-layered regulatory requirements"* (i.e. the complexity induced by the high level of overlapping regulatory project and product requirements);
- *"Large initial investment for third parties"* (i.e. the high threshold for third parties to introduce new products or practices in and for the pharmaceutical industry);
- *"Fast changing regulation and legislation"*;
- *"Old fashioned attitude"* (i.e. old-fashioned attitude of manual and rigorous project governance).

PC2 was split up on the PSs;

- *"Preference of defined logic over empirical logic"* (i.e. the focus on a strict predefined process instead of focusing on graduate procedural improvements);
- *"Requirements traceability"* (i.e. the traceability of software requirement evolution);
- *"Documentative attitude"* (i.e. the comprehensive amount of documentation that is produced during IT projects);
- *"Estimation difficulties"* (i.e. the difficulties of prospective planning);
- *"Risk oriented decision making"* (i.e. the risk oriented discourse within IT projects).

PC3 was split up in the PSs;

- *"Minimal research"* (i.e. the lack of scientific research of the use of agile methods in the context of the pharmaceutical industry);
- *"The lack of embracing change"* (i.e. the conservative attitude of the pharmaceutical industry which the complete opposite of agile's pursue to change);
- *"Absence of project evaluation"* (i.e. the absence of project evaluations during software development projects);
- *"Obligation of traceable compliancy"* (i.e. requirement of tracing processes, decisions and their effect on project deliverables);
- *"Palmed obstruction"* (i.e. refraining procedural change due to the statement that the problems are 'industry wide').

This list of PS is generally similar to the barriers (i.e. of the use of agile in the medical device industry) identified by McHugh, McCaffery and Casey were therefore considered valid for discussion during the expert interviews [13], [14].

The expert interviews have been conducted using Witzel's Problem-Centered Interviewing method [15]. PCI allows for custom tailoring questions (i.e. based on the PSs) to establish an informal atmosphere in which the respondent can tell his/her story instead of engaging in a question-answer process.

The contacted experts were required to be a domain expert of IT projects, have had experience with software validation, and had to be operating in the Dutch pharmaceutical industry. A total of 55 persons have been contacted which lead to 14 interviews. 10 were held face-to-face, 3 were held via the telephone and 1 was conducted via email. The experts ranged from managerial to operational level, and were from large multinational pharmaceutical enterprises and small subcontracted consultancy firms. All interviews have been fully transcribed.

The transcriptions have been used to extract method requirements for the engineering of an agile method tailored for software development projects in the context of the pharmaceutical industry. The extraction process has been conducted using the software product 'NVivo 10' from QSR international. The method requirements have been extracted using the extraction criteria denoted in **TABLE I**.

The method requirement extraction process resulted in 137 statements, requirements, advices and preferences which were used to engineer an agile method.

Method engineering is performed using a variant of the ViewPoint Framework from Nuseibeh, Finkelstein and Kramer [16]. The ViewPoint Framework allows for the creation of a method based on statements from external stakeholders. The framework allows for formal notation of statements (i.e. ViewPoints), the incremental creation of a method (i.e. Method Templates) and focuses on the

TABLE I: Method requirement extraction criteria

Inclusion criteria	**Exclusion criteria**
IC1. A process that is considered a best practice **IC2.** A step that involves the reduction of overhead of IT projects **IC3.** Items that clearly have been noted as potential solutions for the current problems of IT project practices **IC4.** Handlings that are identified as trending, current practice or future practices **IC5.** A handling should be part of the development, implementation or validation of an IT project	**EC1.** Very specific process step that has not proven to be successful consecutively **EC2.** Historical practices that have been proven to be ineffective or inefficient **EC3.** Requirements that define how software features should be defined **EC4.** Handlings that mention retrospective validation or post-project maintenance **EC5.** Practices that involve procedures that are not directly related to tasks in IT projects. i.e. Change Control procedures **EC6.** Statements that have not been transcribed correctly (unclear or incomplete sentence are subject to ambiguity) and therefore lack context

traceability of method elements to the ViewPoints it is based on. Due to article length limitations, the incremental method engineering version (i.e. ViewPoint Framework's 'Template'), the list of method requirements (i.e. based on the ViewPoints) and the references from the method description to its original method requirements are omitted.

III. RESULT: METHOD æ

Method description has been done using a 'origin' chapter (i.e. for providing an introductory description of the method) and categories based on the 4 'core values' of agile, where the first core value (i.e. "*Individuals and interactions over processes and tools*") is split up over two columns as it entails different elements (persons and processes). TABLE II contains interpreted categories for describing the method.

The roles describe the different sets of exclusive responsibilities required to perform certain tasks during the use of the method. The documents define newly introduced document types that replace existing (V-Model related) documents that often are created during IT projects in the pharmaceutical industry. The process describes the steps to be taken to initiate, execute and manage the software development project. In order to increase the understandability of the method, the order of explanation will be; 'process', 'roles', 'documentation'.

A. Origin

Method "æ" has been created as a substitute of the commonly used V-model to unify practices extracted from agile methods (e.g. iterative processes, frequent deliberations, the focus on the tasks with the highest priority) and common practices from IT projects within the pharmaceutical industry (e.g. document review, risk classification, requirements traceability). This approach is directly derived from the agile principle "*our highest priority is to satisfy the customer [pharmaceutical company] through early and continuous delivery of valuable software*" [1]. It is engineered to be used from the perspectives of a software development team and from the project team within the pharmaceutical company by proposing the method

TABLE II: Determination of method description categories

Agile core value	**Interpretation for category**
"*Individuals and interactions*"	Roles
"*Working software over comprehensive documentation*"	Documents
"*Customer collaboration over contract negotiation"* *"processes and tools*"	Processes
"*Responding to change over following a plan*"	Processes

in two perspectives.

Method æ mitigates continuous procrastination by introducing the focus on continuous improvement within the iterations of the project. This is done by enforcing moments in which the teams can discuss efficiency improvements. Method æ supports business benefits by focussing on the most important needs first, and not having to wait on the collection of peripheral requirements as is done in traditional software development processes.

Method æ originates from the Latin character æ, that pronounced as the ' i ' in 'fine', which graphically illustrates the process model.

B. The æ process

The æ process consists of 5 distinguishable phases. 2 can be considered to be agile (i.e. iterative, self-steering teams, short term goal, engaging stakeholders), and 3 of the phases are fully plan-driven. The phases are represented by the 5 D's: 'Definition', 'Development', 'Dispatch', 'Delivery', and the most important one; 'Deliberation'. As the method will be used by two different parties (i.e. the pharmaceutical company and the software developer), it is described in 2 perspectives; the 'pharmaceutical company perspective' and the 'supplier perspective' (i.e. as 'supplier' is often used to refer to software developers). This approach allows for describing the method from an operational point of view, for better understanding of process steps, roles and the use of documents. **Fig. 1** displays the Method æ process model from the pharmaceutical company perspective, and **Fig. 2** displays the Method æ process model from the supplier perspective.

1) The Definition phase

A project starts with a plan-driven 'Definition' phase. The definition phase consists of the creation of a Project Plan (PP) and, if working with an external company for IT development, the mandatory supplier audit. The PP sets the ground principles of the project (e.g. the planning, budget, high overview of the prospected product, roles and responsibilities), but is a living document. The living part is mainly to be found in the sections 'Definition of Done' (DoD) and 'Definition of Compliance' (DoC). They define the acceptance criteria for a finished agile iteration, and serve as checklists for the project team. The project team is hereafter called 'æ team' which consists of members from the pharmaceutical company and from the supplier. The DoD is used by the æ team members working in the Development phase, and the DoC is used in the Dispatch phase. These two elements are subject to revisions once the teams identify unworkable situations or discrepancies between theoretical process descriptions and practical impediments. This will be elaborated later on in the 'retrospective'-task.

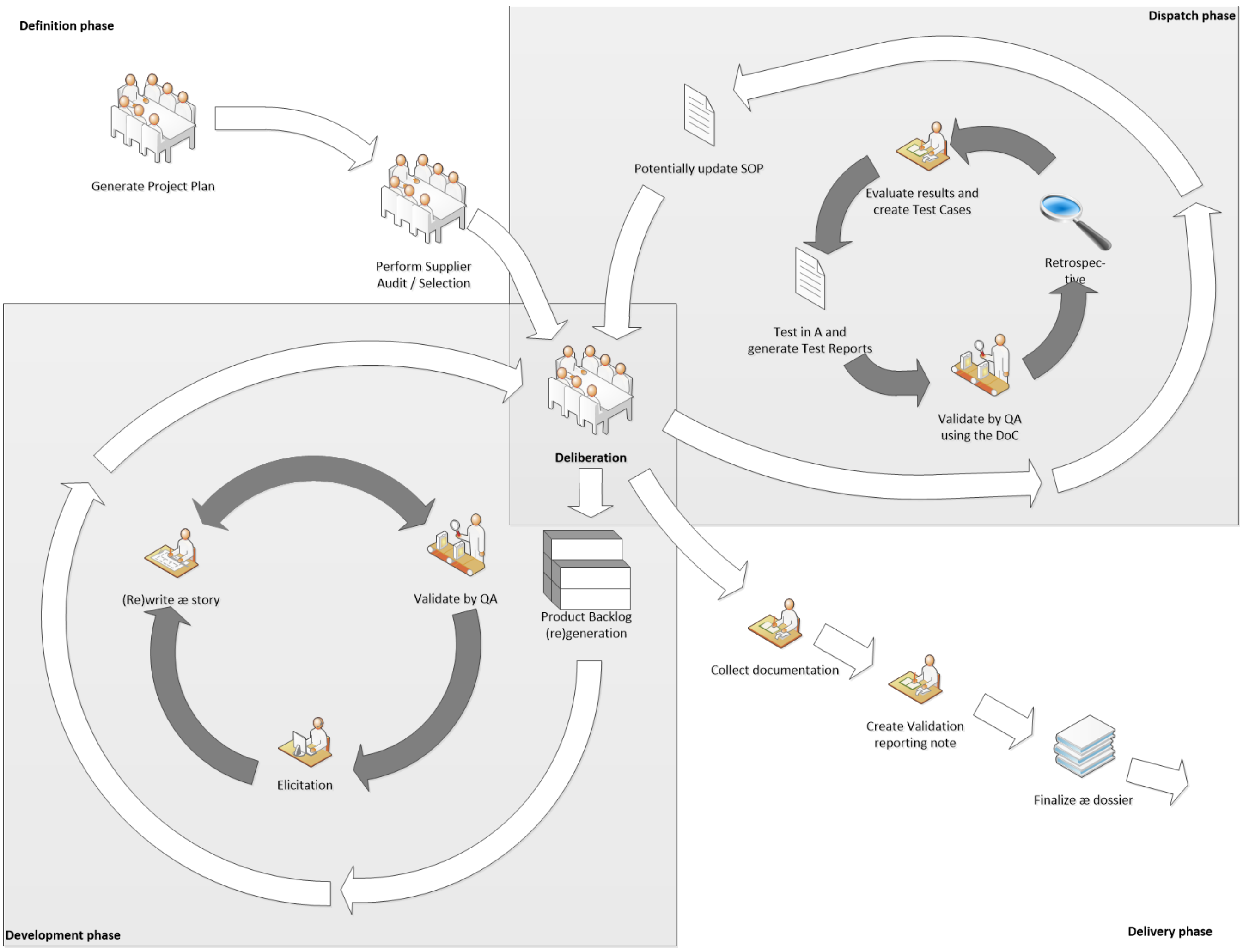

Fig. 1: Method æ process model - pharmaceutical company perspective

The PP is created by the æ team and QA representatives from the pharmaceutical company and identifies the GxP (i.e. Good x Practice) determination of the project. This essentially means that the criticality of the project results (e.g. software, documents, trainings, processes) is assessed using a risk and an impact assessment. A critical GxP project has a high impact on patient safety, medicinal product quality and data integrity. This GxP determination a regulatory obligation [11] and is done by the QA team as it is also the source for the supplier audit.

The Supplier audit is a mandatory QA activity that involves the inspection of a suppliers' Quality Management System. This is done in situations where the pharmaceutical company does not have an internal software development team. During the supplier audit the QA team assesses "*the competence and reliability of a supplier*" so that the "*quality system and audit information relating to suppliers or developers of software and implemented systems*" can be made available to inspectors on request [11]. It is a possibility that the pharmaceutical company has to train the involved team members from the supplier into understanding the practices within the pharmaceutical company (e.g. SOPs, policies and company background).

2) *The Deliberation phase*

The most important phase is positioned in the centre of the model. The Deliberation phase, in which the project related roles are involved in a collaborative meeting to exchange thoughts, requirements, questions and make formal decisions.

The importance of this phase is the fact that project defining decisions are made based on pragmatic reasoning. To elaborate this; the persons that are involved in the meeting are those that do the work (i.e. the æ team), those that define and evaluate risk (i.e. QA representatives) and the one that is in charge of initiating agile processes (i.e. the Product Owner). During this recurring meeting, which is based on Scrum's "*Daily Scrum*" [17], the following elements are discussed:

- 'What has been done in the previous iteration':
    - discuss high risk æ stories;
    - demonstrate and discuss developed product increments;
    - discuss executed test cases and their results.
- 'Agreement of the now':
    - Did the retrospect suggest a project wide change?
    - Does the DoD and the DoC still fit?

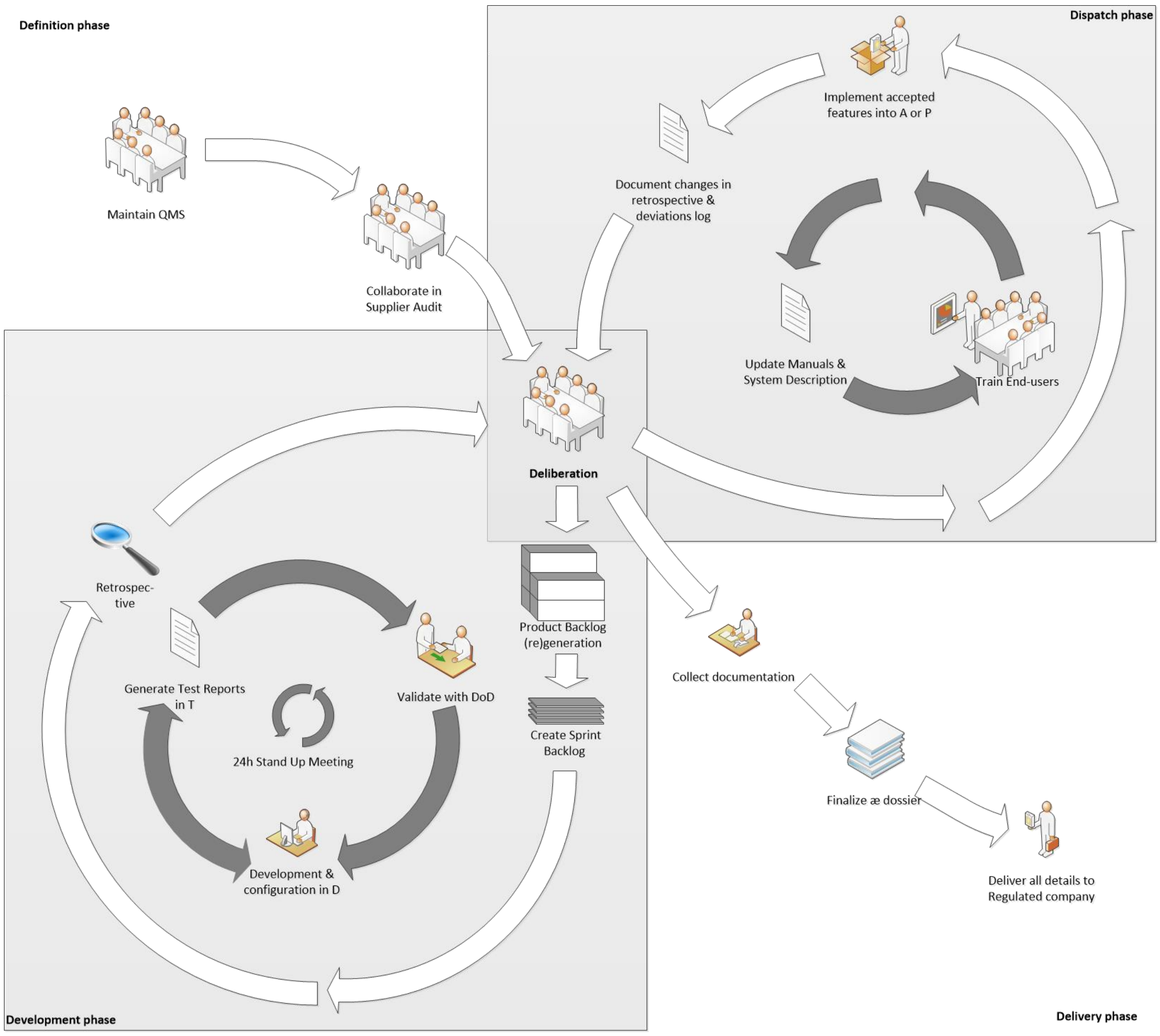
Definition phase
Maintain QMS
Collaborate in Supplier Audit
Dispatch phase
Implement accepted features into A or P
Document changes in retrospective & deviations log
Update Manuals & System Description
Train End-users
Deliberation
Product Backlog (re)generation
Create Sprint Backlog
Retrospec-tive
Generate Test Reports in T
Validate with DoD
24h Stand Up Meeting
Development & configuration in D
Collect documentation
Finalize æ dossier
Deliver all details to Regulated company
Development phase
Delivery phase

Fig. 2: Method æ process model - supplier perspective

  - “Are we on budget?” and thus “Can we continue?”.
  - ‘Priorderize’ (i.e. prioritize and order) the Product Backlog (PBL).
- ‘Elaborate the next iteration’:
  - “Which tests need to be executed again?”,
  - “Is a feature ready to be implemented?”,
  - “What is going to be developed in the next iteration?”

The fact that software is discussed rather than merely documents, it supports the agile principle “*working software is the primary measure of progress*” [1].

All decisions are done collaboratively except for the priorderization of the PBL, as the Product Owner has ownership of this. The decisions made during this meeting are noted in the Retrospective and Deviations Log (RDL) for supporting process traceability. The fact that the priorities are (re)defined at the very beginning of a new iteration, it supports the agile principle “*welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage*” [1]. This creates a situation in which less important requirements remain below the ‘reach’ of the æ team. The longer the low-priority æ stories remain in the list, the filtering (i.e. filter important from unimportant tasks) function has worked. This corresponds with the agile principle “*simplicity—the art of maximizing the amount of work not done—is essential*” [1].

The deliberation meeting can be seen as the roundup of the previous iteration and the start of the next. As multiple phases can be initiated by this meeting, it is possible that the next iteration is performed in 4 dimensions simultaneously:

- development phase from the perspective of the pharmaceutical company, in which software requirements are collected and formally denoted;
- development phase from the perspective of the supplier, in which software is developed;
- dispatch phase from the perspective of the pharmaceutical company, in which the previously developed features are tested;
- dispatch phase from the perspective of the supplier, in which the tested software is implemented.

Technically a 5th process can be initiated together with the aforementioned 4, as the delivery phase is initiated after the successful implementation of developed, tested and approved software.

3) *The Development phase*

The Development phase is one of the agile phases in which there have strict high level boundaries (i.e. iteration length, tasks to be accomplished) but provide the team the freedom on how to organise itself. This mentality is supported by the agile principle "*the best architectures, requirements, and designs emerge from self-organising teams*" [1]. Iterations have a certain timespan (i.e. defined in the PP), which starts and ends with a deliberation meeting. The development phase is described in two perspectives.

a) *Pharmaceutical company perspective*

The æ team is in charge of collaboratively collecting and writing down the software requirements. Requirements are elicited from literature (e.g. SOPs and regulations) and sessions with end-users. But instead of writing down a requirement as a piece of incoherent text, æ team members will write *an* æ *story* (i.e. pronounced as 'a nice story'). The æ story is based on the User Story from eXtreme Programming [18] and replaces the traditional requirement or function specification. The æ story combines the requirement specification, a requirement related risk assessment and acceptation criteria for testing, but is limited to an absolute minimum in size. The rationale behind the delimited and high level notation of the requirements is to engage in meetings and face-to-face contact for feature details, instead of consuming large amounts of time to write down potentially ambiguous software needs.

The æ stories have a strict template in which the team member defines a requirement and performs an initial risk and impact assessment for risks related to patient safety, product quality and data integrity. After finishing the æ story, it is validated by a QA representative. The QA representative has the final say on the risk and impact of a requirement, but can use the advice from the æ team member in making the decision. After this classification, the æ story can be placed in the PBL and is registered in the RTM. Amendment of a requirement therefore requires amendment of a single æ story instead of a complete requirements document.

b) *Supplier perspective*

The æ team at the side of the supplier has the PBL as input. The method of translating the PBL items (i.e. æ stories) into working software is chosen by the team. To oversee the work, it is strongly advised to break the PBL items into smaller tasks that can be executed within a maximum of one day. These tasks can be listed in a Sprint Backlog (SBL). The development phase can be executed as long as the DoD is respected and the mandatory retrospective is performed and in which the RDL is updated. The retrospective is a moment where the team evaluates the last iteration, and decides on whether the DoD fits and whether other quality or efficiency improvements need to be implemented. The retrospective supports the agile principle "*at regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly*" [1].

As all æ stories have a 'story owner', it allows a developer to directly contact the writer of the æ story for details. Because who knows more about an æ story than its author? A few examples are provided of which methods the æ team can employ to do their work.

The Scrum method [17]:

The team starts with a Sprint Planning in which it analyses the work to be accomplished in the upcoming iteration. This is done by examining the Product Backlog and collaboratively calculate the needed time to complete the æ stories. The team always starts at the top (i.e. most important first) and dissects the æ stories into smaller tasks to be completed during the iteration. This is where the SBL is created. "*The Sprint Backlog is a plan with enough detail that changes in progress can be understood in the Daily Scrum*" [17]. The daily scrum is called 'Stand up meeting' in Method æ. The SBL is managed by the team. Every day the team starts with a stand up meeting, which is a 15 minute meeting where every team member explains what has been done yesterday, what will be done today and which impediments have been identified that might threaten finalising the iteration (i.e. according to the DoD). The method of testing is related to the identified risk of the æ story. The higher the risk classification, the more rigorous testing is required. This is described in the DoD.

The Test-Driven Development method [19]:

The test-driven development approach emphasizes the creation of test cases up front. The æ team makes selection from the PBL of the æ stories to be developed in the next iteration. Based on each æ story and the related acceptation criteria, the developments are going to create test cases. This can result in an automated (i.e. test script) or a manual test. The developers will then develop or refactor a feature until the test is executed satisfactory [20]. Together with the æ team members from the pharmaceutical company, tests can be prepared for the acceptance testing in dispatch phase. The test severity will be based on the risk classification of the æ story. Because "*if it's not worth testing, why are you wasting your time working on it?*" [20]. Developed product increments are evaluated on completeness via the DoD. Finished elements and test results are presented during the deliberation meeting.

The eXtreme Programming method [18]:

An XP team starts an iteration with a planning in which an honest selection in made of which æ stories from the PBL are able to be created before the iteration ends. "*Release plans have no details other than a list of stories to be done by a date*" [21] where the release plan is the section of the PBL to be used during the next iteration, and the date is the end of the iteration. The æ team will then develop a plan which "*will often be verified by breaking the stories into development tasks and estimating them with finer grain units*" [21]. This plan can be considered to be the SBL. The team will keep close contact with the story owners for communication regarding to æ story details. Members of the æ team from the pharmaceutical company can be involved closely to be "*the customer on site*" [22]. Daily planning for development, impediment solving and status updates are done by performing the Stand up meeting. “*Communication among the entire team is the purpose of the stand-up meeting*” [22]. Pair programming (i.e. one programming and one evaluating developer sitting next to each other) can be considered to develop software and increase the quality simultaneously [23]. Eventually, the completed software has to be evaluated on completeness using the DoD.

Software development is done in separated programming environments. By using the DTAP approach, development is done in D (development environment), and testing is done in T (testing environment). DTAP is “*a well known industry practice to use multiple similar environments to separate core development from testing and production”* [24]. The other two environments (i.e. acceptation and production) are located at the site of the pharmaceutical company and will be used during the Dispatch phase.

4) *The Dispatch phase*

The second agile phase consists of operations that are focussed on bringing the developed solutions to a compliant state. And therefore sending, or *dispatching* them into practice. The Dispatch phase can therefore be considered to be an after effect of the Development phase. If initiated, the Dispatch phase starts simultaneously with, and has the same length in time as, the development phase. It too has two perspectives; the supplier perspective and the pharmaceutical company perspective. In both cases the æ team needs to respect the criteria denoted in the DoC.

a) *Pharmaceutical company perspective*

The delivered product increments from the development phase, which are presented and accepted in the deliberation phase, are subject to testing in the dispatch phase. æ team members will create test cases on a level that is based on the risk qualification of the related æ stories. The tests are performed in the acceptation environment (i.e. part of the DTAP approach) and test reports are created depending on what is agreed in the DoC. The test results should clearly identify the performed tests, the related æ stories, the test results and whether they meet the acceptation criteria from the related æ stories.

Every test result, with the corresponding test case and the related æ story, is registered in the RTM and is forwarded to the QA team for validation. The QA team has the right to; approve the test and therefore the developed solution, reintroduce the test result as a new æ story to the PBL (i.e. due to the occurrence of a bug, or faulty developed software), or reject the test result. This verdict is done based on the initial risk and impact that has been related to the æ story. These verdicts are registered in the RTM and RDL.

The choice for approving a test result and therefore approving an æ story might require an update of a SOP or a Work Instruction (WI). This process is done in parallel of the IT project.

The Dispatch phase ends in a retrospective in which the pharmaceutical company looks back to the method of testing, validating and implementing deliverables in the last iteration and determine whether anything process related should change (e.g. the DoC). These changes are proposed and initiated during the Deliberation meeting.

b) *Supplier perspective*

Depending on the verdicts from the deliberation phase, product increments (i.e. software features and reports) can be implemented into the production environment. The order of these actions might be different per pharmaceutical company, therefore no fixed procedural descriptions are provided. An example of a process in containing the implementation tasks might be:

The æ team at the supplier first updates the documentation that might be used during trainings (e.g. manuals and system descriptions). A meeting is scheduled to train end-users for the upcoming software update. After the training, the implementation can be performed to release the software increment into an acceptation (i.e. for end-user testing) or the production (i.e. after sufficient and successful testing) environment. These changes and/or encountered issues are documented in the RDL.

5) *The Delivery phase*

The delivery phase is a plan-driven phase in which the Product Owner and the QA representatives wrap up the previous iteration. This phase ensures that after every iteration the project team and therefore the pharmaceutical company is ready for regulatory inspections.

The phase starts by collecting the generated documentation and checking the completeness of it (i.e. do the implemented features have test reports, test cases, æ stories? Is the process traceable using the RTM and the RDL?).

A small summary has to be created to serve as a front page of the on the æ dossier of immediate iteration recognition:

1) Which æ stories are created?
2) Which æ stories have been implemented?
3) Which risks have been identified, and are these mitigated?
4) Which SOP’s and WI’s have been affected during or by the sprint?

The æ dossier can be signed off and archived afterwards.

C. *The æ roles*

Instead of having a granular set of IT project related roles (e.g. 31 role types identified in GAMP5, 9 in EU GMP Annex 11), Method æ has a limited number of roles to reduce confusion and imposed restrictions. Limiting the number of role types increases the level of responsibilities per role, which supports the notion of self-organising teams.

1) *The Product Owner*

The Product Owner role is based on the similarly named role from Scrum. The Product Owner is defined as a person that “*is responsible for maximizing the value of the product and the work of the Development Team*” [17]. However, there are two details that are different in the Product Owner role definition of Method æ; The Product Owner is responsible for maximizing the value and traceability of the product and the work of the æ team. One of his main responsibilities is the right to ‘priorderize’ the PBL. While others are able to advice

or influence the Product Owner, he/she reserves the right to:
1) Enter new æ stories into the PBL;
2) Alter the æ story title, for a better overview of the PBL as a whole;
3) Priorderize (i.e. prioritize and order) the PBL;
4) Ensure the RTM is filled in and is up to date.

2) *The Quality Assurance Representative*

The QA representative is part of the QA team, which consist of persons that have the jurisdiction to evaluate, approve or block æ stories and test results. QA representatives often are related to the Quality Assurance or Quality Control departments within the pharmaceutical company. QA representatives have the responsibility to:
1) Support or perform the documentation tasks for the PP;
2) Support in the development and modification of the DoD and the DoC;
3) Perform the supplier audit;
4) Discuss and identify during the deliberation phase whether æ stories, test reports or developed features have a high risk;
5) Evaluate and determine the risk classification of æ stories and test results;
6) Evaluate the retrospective notes in the RDL, and their execution in practice;
7) Judge the completeness of the æ dossier.

3) *The* æ *team*

The æ team has the most responsibilities, as it is in charge of developing value for the pharmaceutical company. The æ team is an overarching name for the project members from the pharmaceutical company and from the supplier. The reason for combining the teams is trivial; both are busy with turning business needs into company assets. Close collaboration and communication allows for the reduction of time-consuming channelled questioning. This corresponds with the agile principle "*business people and developers must work together daily throughout the project*" [1]. And as æ team members from the pharmaceutical company are able to also 'develop' value (e.g. developing a new SOP), they are as special as those that are able to develop software. And vice versa; æ team members from the supplier are able to create æ stories as well (e.g. after encountering a non-critical bug or a design flaw). The constant strive for improvement, and the ability to do so in the next iteration (i.e. short horizon) allows for achieving the technical excellence that is described by the agile principle "*continuous attention to technical excellence and good design enhances agility*" [1]. The æ team can be considered to be a self-organising team, as it has the room to steer, monitor and operate on itself. This supports the agile principle "*build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done*" [1].

The members of the æ team are responsible for:
1) Working as a team in keeping each other informed and motivated;
2) Developing æ stories using the an æ story template (i.e. becoming æ story owner);
3) Respect, evaluate and discussing the DoD and DoC;
4) Creating test cases, execute tests and create test reports;
5) Collaborate in the retrospective;
6) Maintaining the RTM and RDL.

D. *The* æ *documentation*

Method æ limits the number of documents and document size by using referrals and logs. The PP, æ stories, SBL, test cases, test reports and the RDL are documents that contain distinguishable content. The RTM, PBL and æ dossier are documents with more or less references to the other documents. And due to constantly updating (i.e. expansion), constant reviewing (e.g. during the deliberation meetings) documents do not have to be finished (i.e. written, reviewed and accepted) completely before further use in the process.

1) *Project Plan (PP)*

A Project Plan is key for project success [25]. It allows for the alignment of agreements within multi-stakeholder projects. Method æ version of the PP is a mandatory document as it merges different traditional documents and mandatory sections into one.
1) Project description: The project description explains the business benefits for the pharmaceutical company.
2) Product vision: The product vision describes two elements; the high level overview of the product that is intentioned to be developed, and the position of the product within the company [11].
3) GxP determination: An initial GxP risk assessment has to be done to determine the risk and impact of the to-be-developed system [8]. Based on this verdict, the level of testing and required QA involvement is determined.
4) Roles and responsibilities: The project stakeholders need to be defined [11].
5) Budget: Clear constraints need to be defined in terms of project length, iteration length, reserved money, and available supportive resources (e.g. work location or server capacity).
6) SOPs and contracts: Instead of rewriting sections of procedures and contracts, the PP will refer to the documents that need to be taken into account during the project. This might be the documented process that will impacted by the project (SOP) or the service level agreement with the supplier.
7) Definition of Done: A changing list (i.e. max. 1 A4) with criteria of a finished development iteration such as: "A finished software increment is entirely based on an æ story, tested on failures using the acceptation criteria and is ready to be presented during the deliberation meeting" or "A complete æ story has all elements filled in, is validated and accepted by a QA representative and is traceable in the RTM". The list can be updated after consensus in the deliberation meeting.
8) Definition of Compliance: A changing list (i.e. max. 1 A4) with criteria of delivering regulatory compliant solutions. In essence this list defines different elements which are already embedded in Method æ, such as; "A finished test is performed risk-oriented, consists of QA validated test results, a test case, the related æ story and is traceable in the RTM". The list can be updated after consensus in the deliberation meeting.

2) *An æ story*

An æ story (pronounced as “a nice story”) is the centre piece of Method æ documentation. Instead of the generation of requirements documents and translating this to different formats such as functional designs, the æ story is based on XP’s User Story [18]. The æ story is therefore more than a requirement. An æ story is developed by the æ team and is stored in the PBL. Both æ team members from the supplier and the pharmaceutical company are able to create æ stories. Where æ stories are created by team members from the pharmaceutical company after elicitation with end-users, team members from the supplier are able to create æ stories as a result of a reported bug or required design upgrade. There is no limit in the number of æ stories an æ team member can create.

An æ story has at least the following details:

1) (1…1) A reference number, maintained by the Product Owner;
2) (1…1) A clear, but short title defining the æ story;
3) (1…1) The section “As a <user role> I want <feature description> so that <situation or procedure description>” filled in;
4) (1…n) Acceptation criteria “IF <situation or result> THEN <action> ELSE <action>”;
5) (1…1) A 1-3 rating of impact of the æ story on Process, People, and Technology;
6) (1…1) A 1-3 rating of the risks that might be introduced to Patient Safety, Product Quality and Data Integrity;
7) (0…1) A motivation for the chosen risk and impact levels;
8) (1…1) Name of the æ story owner;
9) (0...n) Notes of the revision history of the æ story.

æ stories are validated by QA representatives, primarily on the risk and impact level. This validation can result in acceptance of the æ story, an alteration of the story’s risk classification or a rejection due to an unresolved risk. Priorities are not defined in æ stories but are defined its position in the PBL. However, the priorderization is influenced by the risk classification of an æ story.

Due to the addition of the name of the æ story owner, the developer is able to contact the correct person for more details and questions.

3) *Requirements traceability matrix (RTM)*

Method æ advocates to embrace but trace change as traditional intermediate documents (e.g. functional specification and design specification) are omitted. The Requirements Traceability Matrix is used and maintained by the æ team, but is validated by the Product Owner. The RTM allows for tracing the evolution and the validity of æ stories throughout the project. Together with the PBL they provide a clear overview of the project progress.

The requirements traceability matrix contains the following elements (i.e. columns):

1)	RTM ID:	generic reference number.
2)	Date initiated:	Moment of æ story creation.
3)	æ story title:	Title of the story.
4)	æ story version:	If the æ story is updated, the version number is increased.
5)	QA verdict 1:	Accept or reject.
6)	QA risk level:	Risk classification of the æ story.
7)	QA rep. name:	Name of the QA representative.
8)	Dev ID:	Traceability ID for the software increments related to the æ story.
9)	TC ID:	Test Case ID.
10)	TR ID:	Test Report ID.
11)	QA verdict 2:	Accept or reject based on the test case and test report.
12)	QA rep. name:	Name of the QA representative.
13)	Delivery date:	Moment of implementation.
14)	Updated delivery date:	Delivery of an updated version (i.e. after an emergency bug fix).
15)	Changes documented:	Y/N on whether software changes have been documented in SOPs or manuals.

Every role has its own responsibilities for filling in columns, where the æ team has: 8, 9, 10 and 15; the QA representative: 5, 6, 7, 11 and 12; and the Product Owner: 2, 3, 4, 13 and 14.

4) *Retrospective and deviations log (RDL)*

EU GMP Annex 11 dictates that “*validation documentation should include change control records (if applicable) and reports on any deviations observed during the validation process*” [11]. GAMP5 also mentions that “*regulated companies should take justified and documented decisions*” (ISPE, 2008). Therefore, in order to trace changes to the product, the RTM is created. Changes in the process are logged in the RDL. The RDL is updated every time a procedure is changed or a decision which causes deviation is made. This log is evaluated by QA and the team, which can result in a DoD or DoC update.

Every record should contain a WHEN, WHAT and WHY (e.g. “on January 1st 2014, the æ story with the ID #123 has been rejected due to not needing the report anymore”). The RDL is not there to pinpoint persons, and does not have to contain names.

5) *Product Backlog (PBL)*

The Product Backlog is “*an ordered list of everything that might be needed in the product and the single source of requirements for any changes to be made to the product*” [17]. The PBL only contains æ stories which have been validated in an earlier stage. The Product Owner has the responsibility to keep the PBL clear (i.e. redefine æ story titles) and is the only one that has the right to ‘priorderize’ (i.e. prioritize and order) the PBL entries.

The PBL is transparent for all project stakeholders. The æ team members that are going to develop software solutions will use the PBL as a prime source of input to plan the iteration and identify the to-be-developed elements of the software. Therefore it is always changing in terms of new additions, altered æ stories or an altered order. The order is discussed during the deliberation meeting, as it will initiate a new iteration in which the æ team at the supplier is going to develop the ones with the highest priority (i.e. highest in the list) first.

6) *Sprint Backlog (SBL)*

The Sprint Backlog is similar to the PBL, but is entirely managed by the æ team at the supplier. At the start of a new iteration, the team collects the æ stories with the highest priorderization and translate these to smaller development tasks. These elements construct the SBL. At the moment of finishing the SBL creation, which is during the first day of the iteration, the SBL does not change. The SBL is the list of work to be done in the next iteration, and therefore only the æ team members that are developing the software can add, remove or order the list elements. The Scrum Guide defines the SBL as a document that visualizes "*all of the work that the development team identifies as necessary to meet the Sprint Goal*" [17]. As Method æ is not working with a Sprint Goal, it is essentially what is needed to be done to finish the iteration.

The SBL is re-created at every new iteration.

7) *Test Cases (TC)*

The test cases do not have a specific format, as it is the responsibility of the team and QA representatives to choose their method of testing and TC setup. However, the level and detail of testing is influenced by the risk classification of the æ story. A test case has to clearly refer to which æ story it is testing. This should be visible on the TC itself as well as in the RTM.

8) *Test Reports (TR)*

Method æ does not dictate the method of creating test reports. This is to the will of the QA representatives. The test results should be in sufficient detail for the QA representatives to evaluate the correctness and completeness of the executed test.

A test report however always has to have the following details registered:

1) Name of the tester (i.e. æ team member);
2) Related test case;
3) Date and time of test execution;
4) Final verdict (PASS / FAIL).

Due to the use of a RTM the level of details on the test reports can be limited. But as QA is validating the test result, the final verdict can be updated by the QA representative. An example is a test case that resulted in a minor pass (i.e. a few errors occurred, but were not identified as a high risk). While the tester had the verdict of a passed test, the QA representative might conceive the induced risk differently. The QA representative has the final say in the outcome of a test result.

9) æ *dossier*

The æ dossier is a collection of the different documents and logs that are brought together at the end of every iteration to satisfy audit purposes. As part of the generation of a holistic project overview, the æ dossier reflects on the previous iteration and describes the compliance with the PP.

The æ dossier is a combination of folders and a 1 page cover that identifies:

1) A reference table of the dossier content;
2) Which æ stories are created? (reference numbers to æ stories);
3) Which æ stories have been implemented? (reference numbers to æ stories);
4) Which risks have been identified, and are these mitigated? (reference to PP risks, æ story risks, and test results);
5) Which SOP's and WI's have been affected during or by the sprint?
6) A signature of the Product Owner and a QA representative.

IV. METHOD EVALUATION AND VALIDATION

Method æ is a tailored method for allowing software development projects in the context of the pharmaceutical industry to benefit from agile practices while remaining compliant to EU regulation. Optimal validation of Method æ would be by applying the method in multiple case studies. But as this is planned for further research, validation is performed by discussing the potential application of the method with a software developing company and a pharmaceutical company. However, a more elaborate evaluation of the method is done in two steps:

- Method æ is compared with the 12 agile principles described in the agile manifesto;
- Method æ is positioned against the documents, processes and roles that are described in pharmaceutical artefacts; GAMP5 [8], PIC/S [26], the EU GMP Annex 11 [11], CFR Part 11 [27], and artefacts used in agile software development; Scrum [17] and XP [18];

A. *Comparison with the 12 agile principles*

An agile method is considered to be 'agile' if it adheres to the agile principles. However, the level of required adherence is not defined and is therefore rather questionable. Dingsøyr, Nerur, Balijepally and Moe mention that agile methods should be "*adhering to varying degrees to the tenets of the manifesto*". These methods "*include eXtreme programming (XP), scrum, lean software development, feature-driven development (FDD), and crystal methodologies, to name but a few*" [2]. By mentioning "*varying degrees to the tenets of the manifesto*" and summing up a list of the most well-known agile methods, an assumption can be made that these methods do not adhere to the 12 principles fully.

Method æ has been compared with the 12 agile principles, which resulted into 3 questionable links:

- Principle 3: "*Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale*" [1].
  Method æ does not mention the optimal length of an iteration due to the absence of a practical case study. Therefore no indisputable conclusion can be made regarding to this principle. The Scrum Guide clearly mentions "*Sprints are limited to one calendar month*" [18], while XP mentions that its iterations should be "*1 to 3 weeks in length*" [21].
- Principle 6: "*The most efficient and effective method of conveying information to and within a development team is face-to-face conversation*" [1].

No method requirement mentions the communication method for working on IT projects with two potentially geographically distributed teams (i.e. pharmaceutical company and supplier). Where Scrum has the Product Owner (i.e. representative of the 'customer' that works in close collaboration with the development team, XP has the 'on site customer' for its availability for direct questioning by the development team. Due to the absence of a case study, no definitive answer (i.e. on how to optimally engage cross role and company communication) can be provided for Method æ.

- Principle 8: "*Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely*" [1].
  Generating a "*constant pace*" allows or adaptive prospective planning. However, working with æ stories which essentially are high level requirements, it is difficult to inductively reason based on Scrum or XP practices on how a certain pace is maintained. Method æ does not mention how this constant pace is maintainable as it requires practical implementation (i.e. case study) and empirical observation to generate a definitive answer on this principle.

B. *Positioning against industry standards, regulation and agile methods*

A comparison has been done to evaluate whether Method æ has an intended overlap and therefore is a substitute to existing and traditional software development projects in the pharmaceutical industry. The comparison has been done on processes, roles and documents that are described in:

- 2 industry standards for IT projects in the pharmaceutical industry which were identified during the expert interviews as the ones that are used most often; ISPE's GAMP5 [8] and PIC/S's "*Good practices for computerized systems in regulated "GxP" environments*" [26];
- 2 regulations that were discussed the most often during the expert interviews; EU GMP Annex 11 [11] and FDA's CFR Part 11 *"Electronic records; Electronic signatures"* [27];
- 2 most popular agile methods according to Dingsøyr, Nerur, Balijepally and Moe [2]: Scrum [17] and XP [18].

ISPE's GAMP5 is written in the broadest fashion, potentially covering all situations that can occur during a software development project in the context of the pharmaceutical industry. Therefore, due to its broadness, it has been used as the initial list of processes, roles and documents. Every row identified another artefact, role or process. Similar descriptions resulted in positioning the artefact in the same row of the existing (i.e. written) ones. Therefore the titles in a single row can be different (e.g. GAMP5's "Repair activity" is similar to PIC/S' "Emergency Changes"). Elements that were not related or similar to Method æ are identified to be 'peripheral IT project documents, processes and roles'. The following examples will explain the absence of a relation or similarity:

- Method æ does not inhabit a 'Data Migration Plan' or a 'Data Migration Report', where GAMP5 does. These documents can be part of an IT project, but are not mandatory elements of an IT project. Method æ is described on a generic level in which these types of documents are not implemented. This is the same for GAMP5's and PIC/S' 'Validation Master Plan'. Method æ does inhabit validation policies in its Project Plan, but does not formally define the creation of a separate document.
- GAMP5 and CFR Part 11 identify 'Document Management' processes and 'Adequate controls for documentation'. Both describe how to maintain documentation practices, where Method æ does not inhabit this.
- GAMP5 and PIC/S identify the role 'auditor', where GMP Annex 11 describes 'Inspectors'. Both descriptions explain post-development validations performed by external parties to assess the level of compliancy to SOP's and regulation. Method æ is a method that is focused on the practices and persons that are involved in the IT project itself. The method enforces (via its process steps and documents) compliancy.

Therefore Method æ cannot be considered to be a system development lifecycle approach (SDLC). An SDLC "*entails defining and performing activities in a systematic way from conception, understanding the requirements, through development, release and operational use, to system retirement*" [8]. Method æ is fundamentally different in the last two sections of the SDLC definition described in GAMP5. Method æ does not focus on the operational use and retirement of a computerised system.

C. *Method validation*

Method æ has been discussed with 2 market representatives. The initial interview has been conducted with the managing director of a software development company from Utrecht. This company is specialized in insurance, banking and pharmaceutical software. The second interview has been conducted with the head of Quality Assurance of a pharmaceutical company located in the city of Nijmegen that is specialized in monitoring patients that are on high risk medication and is a Marketing Authorisation Holder of a generic medicinal product.

The managing director identified the following considerations that have to be made upon implementation:

1) The risk of risk identification in an æ story (i.e. combining multiple low-risk æ stories might induce a high risk);
2) Self-organising teams require support and strict guidance (i.e. as an experienced coach of self-steering teams, the managing director emphasizes the time and effort required to get a team to a self-organising mentality);
3) The use of living documents need to be enforced (i.e. the living documents are key, and therefore need to be enforced via tools);
4) Method æ keeps the door open for use of development methods (i.e Method æ allows for different use of methods within the software development team itself. This freedom is "*a big plus*");
5) Method æ should prove itself (i.e. a new method always requires early adopters. Once it is proven, others can piggyback).

The head of quality assurance identified the following considerations and concerns:

1) Method æ should become the primary process, with no further discussions (i.e. such process model will only be successful once it has managerial support and therefore is enforced);
2) Method æ will require more involvement of QA, but ensures IT project efficiency (i.e. the frequent meetings and short cycle review moments require QA involvement more often. But due to this close involvement, QA is more acquainted with the project details);
3) The living documents need to be maintained (i.e. the reduction of document types means there is a higher dependability on the living documents);
4) A certain 'internal audit' step is missing (i.e. the lack of an inter-project procedural check is missing, although the head of QA mentions the possibility of using of the æ dossier for intermediate auditing).

V. Conclusion

Software development projects in the pharmaceutical industry are subjected to a high level of governance in terms of process control, process traceability and risk oriented decision making. Generic agile methods do not entail these practices by default, resulting in the absence of agile adoption in the pharmaceutical industry.

This research is conducted to identify the software development project-related needs from the industry and to engineer a tailored method to fit these needs and benefit from agile methodologies.

Method æ has been engineered by combining 137 method requirements extracted from 14 expert interview transcriptions. Method æ introduces iterative software development in which risk oriented decision making, cross-company collaboration and document minimization predominate. As it is not considered a Software Development Life Cycle but a software development method for managing the software development process, it serves as a substitute to traditional, slow, highly documentative models such as the V-Model.

VI. Discussion and Further Research

The lack of a case study has its impact on providing definitive statements and answers. Method æ therefore lacks the following elements;

- Method æ is not created for large scale projects. Large projects have impact on the complete organisation and often require years of execution adhere to different perspectives, whereas Method æ is focussed on small projects for developing customized software products. To reduce ambiguity; small can be perceived as less than ½ year. In theory, the method should be usable for a longer time while maintaining a constant pace of development. However, this has not been proven.
- Method æ and the related research have not been focussed sufficiently on the adoption of an agile method by a high-governance company. The change management that is related to this research limitation has been well described in scientific articles for other industries. Future research might involve the generalisation of the knowledge of agile adoption in other industries and apply it in the pharmaceutical industry using Method æ.
- Larger teams often operating in multiple projects that run simultaneously. No research has been done on what the effect is of running multiple IT projects with Method æ simultaneously while remaining the same team for multiple projects.

The use of æ stories is largely different from the use of a Requirements Document in which all requirements are registered, connected and validated. Single æ stories have individual risk assessments while multiple low risk æ stories can collectively induce a high risk. Further validation by more quality oriented experts and a case study might identify practices to mitigate or at least identify these inter-æ story risks.

References

[1] K. Beck, M. Beedle, A.V. Bennekum, A. Cockburn, W. Cunningham, M. Fowler, et al. (January 2014). Manifesto for Agile Software Development. *Agilemanifesto.org* [Online]. Available: http://agilemanifesto.org/

[2] T. Dingsøyr, S. Nerur, V. Balijepally, and N. Moe, "A decade of agile methodologies: Towards explaining agile software development," *The Journal of Systems and Software*, vol. 85.6, pp. 1213-1221.

[3] B. Fitzgerald, K. Stol, R. O'Sullivan, and D. O'Brien, "Scaling Agile Methods to Regulated Environments: An Industry Case Study," *Proceedings of the 2013 International Conference on Software Engineering*, pp. 863-872, San Francisco: IEEE Press, 2013.

[4] L. Heeager and P. Nielsen, "Agile Software Development and its Compatibility with a Document-Driven Approach? A Case Study," *20th Australasian Conference on Information Systems*, pp. 205-214, Melbourne, 2009.

[5] S. Carleysmith, A. Dufton, and K. Altria, "Implementing Lean Sigma in Pharmaceutical research and development: a review by practitioners," *R&D Management*, vol 39.1, pp. 95-106, 2009.

[6] M. McHugh, F. Mc Caffery, and V. Casey, "Integrating Agile Practices with Plan-Driven Medical Device Software Development," *13th International Conference on Agile Software Development*, Vienna, Austria, June 2012.

[7] M. Awad, "A Comparison between Agile and Traditional Software Development Methodologies," Dissertation, Dept. CS. and Soft. Eng., University of Western Australia, Crawley, Australia, 2005.

[8] *GAMP5 - A Risk-Based Approach to Compliant GxP Computerized Systems*, ISPE, Tampa, FL, 2008.

[9] M. McHugh, O. Cawley, F. McCaffery, I. Richardson, and X. Wang, "An Agile V-Model for Medical Device Software Development to Overcome the Chllenges with Plan-Driven Software Development Lifecycles," *5th International Workshop on Software Engineering in Health Care*, pp. 12-19, IEEE, 2013.

[10] A. Hajou, “Shaping agile software development methods for the highly documentative pharmaceutical industry,” M.S. thesis, Dept. Inf. CS., Utrecht University, Utrecht, the Netherlands, 2014.

[11] *EudraLex - Volume 4 GMP Annex 11: Computerized Systems,* European Commission - 2011

[12] A. Hajou, R.S. Batenburg, and S. Jansen, “How the Pharmaceutical Industry and Agile Software Development Methods Conflict,” *14th International Conference on Computational Science and Its Applications*, pp. 40-48, Guimarães, Portugal: IEEE, 2014.

[13] M. McHugh, F. McCaffery, and V. Casey, “Barriers to Adopting Agile Practices when Developing Medical Device Software,” *Software Process Improvement and Capability Determination*, pp. 141-147, 2012.

[14] M. McHugh, F. Mc Caffery, and V. Casey, “Barriers to using Agile Software Development Practices within the Medical Device Industry,” *European System, Software & Service Process Improvement & Innovation*, Vienna, Austria, 2012.

[15] A. Witzel, “The Problem-Centered Interview,” *Forum: Qualitative Social Research*, vol. 1.1, 2000.

[16] B. Nuseibeh, A. Finkelstein, and J. Kramer, “Method engineering for multi-perspective software development,” *Information and software technology,* vol. 38.4, pp. 267-274, 1996.

[17] K. Schwaber and J. Sutherland, “The Scrum Guide,” Scrum.org, 2013.

[18] K. Beck, *Extreme Programming explained: Embrace change,* Boston, MA: Addison-Wesley Professional, 2000.

[19] K. Beck, *Test-Driven Development: By Example,* Boston, MA: Addison-Wesley Professional, 2003.

[20] S. Ambler. (May 2014). Introduction to Test Driven Development (TDD). *Agiledata.org* [Online]. Available: http://www.agiledata.org/essays/tdd.html

[21] D. Wells. (May 2014). Iterative. *Agile-process.org* [Online]. Available: http://www.agile-process.org/iterative.html

[22] D. Wells. (May 2014). Daily Stand Up Meeting. *Extremeprogramming.org* [Online]. Available: http://www.extremeprogramming.org/rules/standupmeeting.html

[23] J. Grenning, “Launching Extreme Programming at a Process-Intensive Company,” *IEEE Software*, vol. 18.6, pp. 27-33, 2001.

[24] I. Heitlager, S. Jansen, R. Helms, and S. Brinkkemper, “Understanding the dynamics of product software development using the concept of coevolution”, *Second International IEEE Workshop on Software Evolvaility*, pp. 16-22, Philadelphia, PA: IEEE, 2006.

[25] J. Verner and N. Cerpa, “Australian Software Development: What Software Project Management Practices Lead to Success?” *Australian Software Engineering Conference*, pp. 70-77, Sydney: IEEE, 2005.

[26] *Good practices for computerized systems in regulated "GxP" environments,* Pharmaceutical Inspection Co-operation Scheme, Geneva, Switzerland, 2007.

[27] *Code of Federal Regulations, Title 21, Chapter 1, Subchapter A, Part 11: Electronic Records; Electronic Signatures,* U.S. Food and Drug Administration, Silver Spring, MD, 2013.