CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Introducing traceability in an agile development environment

Using a systematic methodology for designing and implementing a tailored traceability strategy at 1928 Diagnostics

Master's thesis in Computer science and engineering

GABRIEL ORSTADIUS

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2020

Master's thesis 2020

Introducing traceability in an agile development environment

Using a systematic methodology for designing and implementing a tailored traceability strategy at 1928 Diagnostics

Gabriel Orstadius

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2020

A Chalmers University of Technology Master's thesis template for LaTeX
Using a systematic methodology for designing and implementing a tailored traceability strategy at 1928 Diagnostics
Gabriel Orstadius

Supervisors: Salome Honest Maro & Jan-Philipp Steghöfer, Department of Computer Science and Engineering
Advisor: Robert Engberg, 1928 Diagnostics
Examiner: Michel Chaudron, Department of Computer Science and Engineering

Master's Thesis 2020
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2020

A Chalmers University of Technology Master's thesis template for LaTeX
Using a systematic methodology for designing and implementing a tailored traceability strategy at 1928 Diagnostics
Gabriel Orstadius
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Software traceability is the ability to interrelate software artifacts, e.g. work products and requirements, and use these relationships to support different software and systems engineering activities such as ensuring product conformance and conducting change impact analysis. Additionally, several regulatory agencies have explicit requirements regarding traceability. To introduce effective traceability, a traceability strategy tailored for the organization's development process and traceability goals must first be in place, which organizations often struggle with. This action research study aims to show how TracIMo, a methodology for systematically introducing traceability strategies, can be used to design and deploy a tailored traceability strategy. This is done by documenting how TracIMo was used at 1928 Diagnostics, an agile bioinformatics start-up developing infection control products for hospitals, and discussing the benefits, drawbacks, costs and challenges in doing so.

Keywords: Traceability strategy, TracIMo, Agile development environment.

Acknowledgements

I would like to thank my supervisors, Salome Honest Maro and Jan-Philipp Steghöfer for providing a steady stream of valuable advice and guidance. I specially want to Salome for all the feedback on the report. Their support and encouragement was much appreciated. I would also like to thank Michel Chaudron for being my examiner and the feedback he gave during the half-time report.
I also want to thank people at 1928 Diagnostics. Special thanks to Robert Engberg for leading me in the right direction and checking in on me throughout the study, Fredrik Dyrkell for all the informative discussions, and Dimitrios Arnellos for his patience explaining the bioinformatics process, a very enjoyable collaboration and an amazing job developing the traceability tools. All three supported me more than I could have anticipated. I also want to thank Emil Karlsson for his valuable input and everyone at 1928 Diagnostics for being so welcoming.

Gabriel Orstadius, Gothenburg, October 2020

Contents

List of Figures

List of Tables

1

Introduction

Traceability is the ability to interrelate software artifacts, i.e. any unit of data involved in a software development process such as work products and requirements, and is used to support different software and systems engineering activities [4]. Two of the main benefits of traceability are facilitating program comprehension [7] and supporting change impact analysis by helping developers understand how a proposed change will affect the current system [21]. This can facilitate effort estimation and team coordination [4]. Another important use of traceability is for verifying that specified requirements are fulfilled and maintained [4], especially if external regulations must be complied to. Many regulatory agencies of different industries, such as the US Food and Drug Administration (FDA) and US Federal Aviation Administration (FAA), have explicit requirements regarding traceability [8]. For example, safety-critical systems such as medical devices must use traceability to show that product requirements and identified risks are addressed and verified [8].

When traceability is implemented haphazardly or through ad hoc efforts, which often is the case, it is rarely effective [10] [4]. A general reason why effective traceability is difficult to introduce is because it must be tailored to the organization's development process and traceability goals while keeping down the cost of implementing and maintaining it [4]. To implement an effective traceability solution within an organization, a *traceability strategy*, including decisions regarding the creation, maintenance and use of traces, must first be in place [4].

Several studies have been conducted describing the challenges of introducing traceability in various kinds of organizations, including agile organizations. As an example, Cleland-Huang [4] describes how traceability in agile environments should be light-weight. Although there is literature defining traceability assessment procedures such as Rempel et al.'s [11], there is a lack of explicit guidelines on how to systematically design and implement a tailored traceability strategy [9].

This gap is addressed by a methodology called Traceability Introduction Methodology (TracIMo) [2]. TracIMo aims to help companies systematically introduce a traceability strategy by defining concrete steps for eliciting the company's goals, designing a solution accordingly, and lastly deploying and evaluating the solution empirically. However, this methodology has only been evaluated once in a company with an agile development team and therefore this study serves as a second evaluation of the TracIMo methodology.

1.1 Statement of the Problem

1928 Diagnostics is a bioinformatics company developing a cloud-based platform used to help hospital with infection control. The company currently doesn't develop medical devices but they think traceability could be beneficial for developing them in the future. However, like many agile organizations, they lack a traceability strategy [6]. Nonetheless, since rapid development of their current products is a high priority, they do not want to introduce traceability where they do not see very clear benefits that outweigh the cost of introducing and adhering to the traceability strategy. Another dilemma is that they do not know where to start. The company consists of a single development team but developers work on different areas of the platform with different kinds of artifacts and development processes, each of which would need different traceability strategies.

1.2 Purpose of the Study

The purpose of this study is to investigate how to systematically design and introduce an effective traceability strategy in an agile environment. This will be done by using the methodology called TracIMo at 1928 Diagnostics to design and deploy a traceability strategy suitable for the company. The study aims to capture new insights for both researchers and practitioners interested in the traceability strategy introduction process.

1.3 Research Questions

The main research question is as follows:

- **RQ 1:** How can TracIMo be used to introduce a traceability strategy in an agile environment?

To answer this question, the following sub-questions will be answered:

- **RQ 2:** What are the challenges when using TracIMo in this context?
- **RQ 3:** What are the benefits and drawbacks of the designed traceability strategy in this context?

1.4 Contribution

This study gives a detailed account on how a traceability strategy is systematically introduced in an agile environment, something there is a scarcity of in existing literature and companies tend to have little knowledge about [6] [10]. In addition, using TracIMo to successfully introduce a traceability strategy in another company contributes to validating TracIMo's generalizability.
The study is thus of particular interest to researchers and practitioners interested in how to use TracIMo on a company with similarities to 1928 Diagnostics, a small start-up using agile practices.

2
Related Work

TracIMo [2] is based on an assessment procedure introduced by Rempel et al. [11] and builds upon it by refining it and adding steps, making it suitable not only to analyze a company's actual traceability needs, but also as a guide on how to implement a traceability strategy from scratch. Unlike the assessment procedures introduced by Rempel et al., this methodology does not focus on how to assess existing traceability practices but on how to introduce a suitable traceability strategy. The paper also gives an account of how TracIMo was deployed in an agile development team.
Alongside Rempel et al., there is a range of literature covering documented benefits and challenges related to introducing traceability. Of interest for this thesis, some have focused on companies that use agile practices. However, as with Rempel et al., those that produce concrete practical recommendations mostly focus on the assessment of existing traceability practices, some of which are mentioned below.
Cleland-Huang [4] discusses how traceability can be implemented in agile projects. She concludes that despite the de-emphasis on documentation, agile projects still stand to benefit from implementing a lightweight traceability strategy, especially in larger, more complex or safety-critical agile projects.
Mäder et al. [8] evaluate ten companies in safety-critical environments and go through best practices in traceability, common traceability problems, and remedies for how to solve these problems, many of which are applicable to non-safety-critical environments as well. How these recommendations work in combination with agile practices is not discussed.
A study that is related to this thesis is the bachelor thesis conducted by Golemshinka and Kamsheh [3] at 1928 Diagnostics in 2019. In their study, they assessed SafeScrum practices, an adaptation of Scrum intended for development of safety-critical systems [5] against medical device standard requirements and the needs of 1928 Diagnostics. After conducting their analysis, they then collaborated with the company to develop a plan on how to implement a fitting version of SafeScrum. In accordance with SafeScrum, the plan emphasized the importance of traceability but did not provide details on how to implement an improved traceability strategy, other than the introduction of a new role responsible for various activities including *traceability analysis*. The purpose of this activity was to confirm that different outputs of development are traceable. SafeScrum does not go into details on how to develop a traceability strategy either but suggests that trace granularity should be up to the company [5], which is in line with TracIMo. SafeScrum also goes into how a set of coupled tools and agile practice may be combined to realize *living traceability* which greatly facilitates continuous compliance with standards. However, 1928 Diagnostics has not yet implemented any of the planned SafeScrum practices since they

currently prioritize rapid development of the platform and development of medical devices has not been their focus.

3
Background

In this chapters, key concepts needed to understand this thesis are explained. In addition, an introduction to the company 1928 Diagnostics is given.

3.1 Traceability

In this section, traceability concepts used in this thesis will be explained as defined in the book *Software and Systems Traceability*[4]. In addition, documented traceability benefits that can be seen as reasons for agile projects to introduce traceability are presented.
Traceability is the ability to interrelate software artifacts with *trace links*. A trace link is an association between a pair of artifacts that the user should be able to *trace*, i.e. navigate to go from one artifact to the other. Trace links can have various shapes and forms, for example, trace links may be labeled with the *link semantics*, which describes the purpose of the trace link or the nature of the relationship between the artifacts, e.g. "tests". Trace links can also be created and maintained with a wide range of tools, including tools specifically made for traceability, management tools including features that allow for traceability, and adequately configured general-purpose tools such as Excel or databases [15][4]. Trace links are said to be directional, meaning that there is a primary direction for the user to trace the trace link. Depending on the implementation, it may or may not be physically possible to trace a trace link in the opposite direction, in which case the trace link is said to be *bidirectional*.
Two artifacts don't need to have a direct trace link between them for a user to be able to trace from one artifact to the other one. If there is a *trace path* (this is the term used by TracIMo, the book uses *chained trace* instead) connecting the two artifacts, i.e. a path of several traces strung in sequence, then a user can use this trace path to do so by going via several other artifacts.
As briefly mentioned in the introduction, a traceability strategy is a collection of decisions regarding the creation, maintenance and use of individual trace, resulting in a traceability solution that can be implemented. To elaborate further, deriving and implementing a traceability strategy means; deciding which traces to include, deriving *traceability processes* that define how traceability it to be created, maintained, and used, attaining tools for completing the traceability process, and deploying the resulting traceability solution.
When deciding which traces to include, it can be useful to derive a *Traceability information model* (TIM). As explained in the book: *"The TIM is an abstract expression*

of the intended traceability for a project". In other words, a TIM illustrates which trace links between which artifact types are meant to be permissible and supported by the traceability solution. A TIM may also display further details about the desired trace links, e.g. the trace link semantics, or the numeric relationship between linked artifacts. Deciding which traces to implement is an important part of the traceability strategy since each trace link takes effort to create. A TIM should thus be goal-oriented and only include meaningful trace links that are truly needed.
Appleton [21] [4] lists six kinds of traceability benefits that can be reasons to introduce traceability even in agile projects.

1. **Product conformance**, also known as requirements validation, means ensuring that the product fulfills all requirements. Trace links can help to show that each requirement has been fulfilled.
2. **Process conformance** means ensuring that procedural activities, such as tests and reviews, have been conducted. Traceability can help to show that such activities have been conducted by adding trace links to something proving that these activities were conducted.
3. **Change impact analysis** is for analysing what impact a proposed change would have on other parts of a system, which is useful for estimating the involved effort and coordinating with those affected by the change. Trace links can facilitate the analysis by making dependencies more visible.
4. **Project accountability** means ensuring that all changes are for specified requirement and not excess functionality. By enforcing that all code changes are traced to a requirement, this can be avoided.
5. **Baseline reproducibility** means ensuring that all configurations are saved so that relevant system states, i.e. baselines, can be reproduced. This can easily be accomplished with a version control system such as git, although there can be scenarios where saving all configurations within a repository can be challenging. In such scenarios, trace links to artifacts containing the configurations can be used.
6. **Organizational learning** refers to documenting rationales behind critical decisions such that knowledge can be rediscovered. Trace links can help to make these documentations more accessible.

3.2 Biology and Bioinformatics

Since this study is aimed towards software engineers but knowledge in biology and bioinformatics is required to understand 1928 Diagnostics' products, a quick introduction to some important concepts is here given. Information used in this section is mostly taken from resources created by 1928 Diagnostics for introducing new employees (and master thesis students) to these concepts.
DNA's genetic information, including all genes, is stored in a *sequence* of units called *nucleotides*. A *genome*, is an organism's complete DNA sequence. Note that although organisms within a species have many genes in common, their genomes still differ. To obtain a sample's DNA sequence, it must be *sequenced* by a *DNA sequencing machine*, which does multiple reads of different segments of the DNA sequence and then stitches these segments together.

Bioinformaticians, who are knowledgeable about biology, computer science as well statistics, can then create programs to analyze the DNA sequence and reach conclusions. For example, they can answer which bacterial species are in a sample, if they are pathogenic, i.e. capable to cause a disease, and in that case what kind of antibiotic resistances the bacteria in this particular sample has. Just as important, a bioinformatician can also assess the quality of the DNA sequencing, which is relevant since sequencing errors may lead to the wrong conclusions being made.

3.3 1928 Diagnostics

1928 Diagnostics is a bioinformatics start-up with a development team consisting of four software engineers, four bioinformaticians and led by the Chief Technology Officer (CTO). The company started with the mission to combat pathogen's antibiotic resistance, one of humanity's greatest threats, with the power of bacterial DNA analysis.

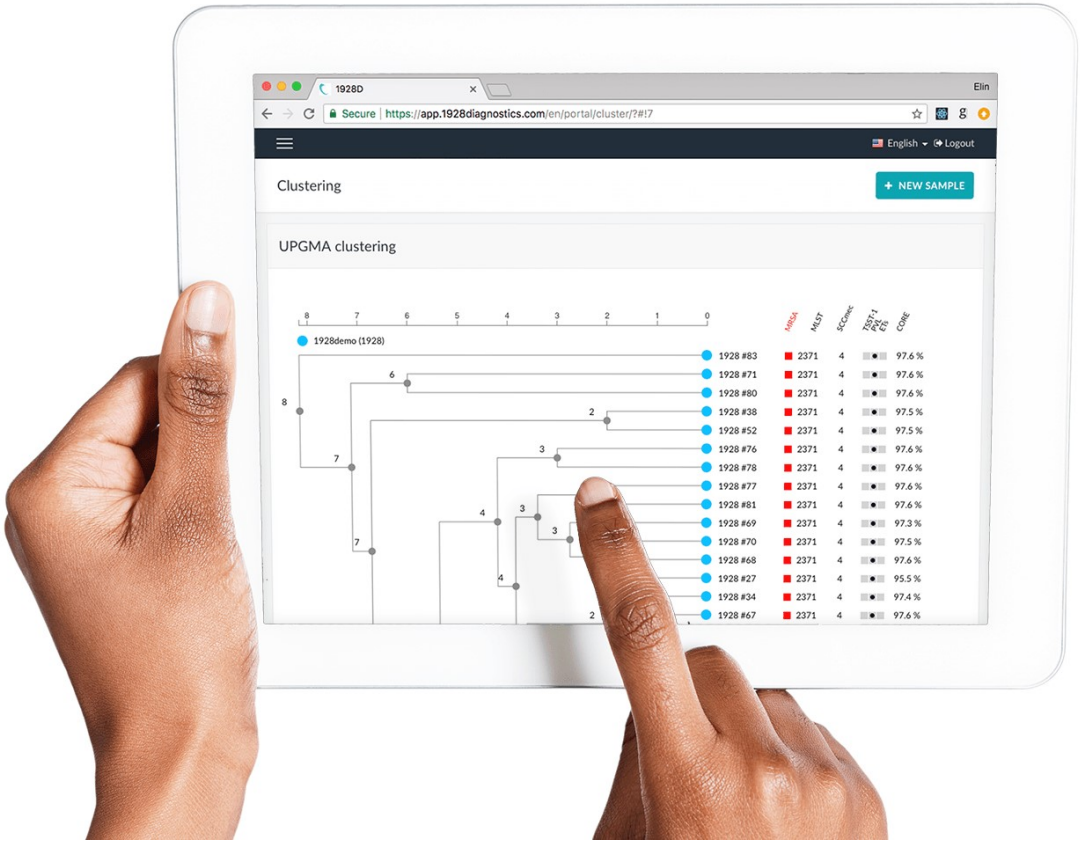
Today their cloud-based platform offers various infection control products that can be used for different kinds of analysis, including antibiotic resistance typing and outbreak tracing. The process for using their products starts with the customer taking samples from patients, putting them in a DNA sequencing machine that outputs a digital file containing the DNA sequence, and then uploading it to 1928 Diagnostics's cloud platform. Then, depending on the pathogen, the kind of DNA sequencing machine used and the kind of analysis the customer wishes to do, different *pipelines* are initiated. A pipeline is a series of automatized steps where the DNA sequences are analyzed that ends with the results being presented to the customer. The customer can then use the results to be more informed on how to act.

Using their platform, a hospital can for example trace the outbreak of an infection occurring within the hospital, thereby helping them stop and prevent further spread. The way it works is that the platform analyzes and clusters the samples based on genetic differences, resulting in a visualised branching diagram, called a *phylogenetic tree*, where the relationships between samples can be further analyzed. Outbreaks can thereby be identified where groups of closely related samples are found as seen in Figure 3.1 [16].

In the future, 1928 Diagnostics may be interested in offering diagnostics tools, i.e. medical devices intended for diagnosing patients. For example, by profiling a pathogen's antibiotic resistance, a doctor can more accurately choose the right antibiotics for a patient. However, before 1928 Diagnostics can sell a widely used diagnostics tools, DNA sequencing machines need to get cheaper and faster. Furthermore, due to strict medical device regulations, developing and certifying diagnostics tools is not easy. In general, medical device regulations emphasize documentation of one's development process. The development process must also adhere to standards, which include traceability and change management aspects.

Today there are other regulations that the development team has to take into consideration, the Health Insurance Portability and Accountability Act (HIPAA) being one of the most relevant ones. HIPAA is an American regulation stipulating how patient information is to be protected. Unlike medical device regulations, HIPAA does not have strict rules regarding the development process or explicit traceability

Figure 3.1: Image taken from 1928 Diagnostics' website showing a phylogenetic tree [16].



requirements, only specific requirements on how to protect patient information that must be fulfilled. These requirements include encrypting data and implementing security measures so data can't be tampered with without detection and have already been fulfilled. However, they would still like to implement proper verification for these requirements, e.g. in the form of tests, and traceability from the requirements to corresponding verification. This would thereby remove the risk of accidental compliance breach, but currently there is no such verification nor traceability. Although implementing verification would just require for developers to create test, the real challenge lies in implementing a suitable process for storing the requirements and creating trace links to the tests.
Due to medical device regulations and other regulations such as HIPAA, 1928 Diagnostics is interested in introducing a traceability strategy. The company does not expect a traceability strategy that would help them comply with regulations right away, but rather something they could build upon in the future. However, a precondition is that any changes to the ongoing development process are cost-beneficial today, regardless if the resulting development process turns out to be fitting in the future. In other words, the company does not want to make sacrifices that would hurt the development of infection control products only for the prospect that these sacrifices will bear fruit. Ideally, they would want to introduce a traceability strategy that both supports the development of infection control products as well as prepares them for the future development of diagnostics tools.
The company's current development process is not dependent on requirements being specified and there is no standardized requirements management system. This makes it much more challenging to introduce a traceability strategy where the verification of requirements can be traced. Another dilemma is that they are not sure which development area to start with. Although the company consists of a single development team, developers work on different areas of the platform with different kinds of artifacts and development processes, each of which would need different traceability strategies.

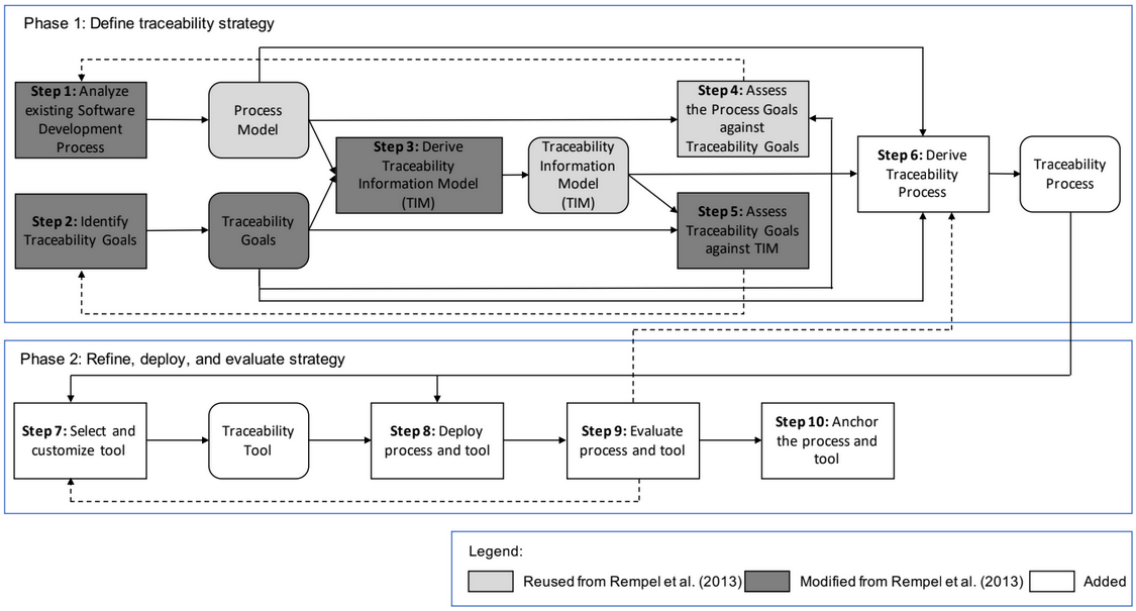
4
Methodology

In this chapter, the methodology used to answer the research questions of this thesis is presented. First, general descriptions of the research methods are presented. Thereafter, how these methods were utilized in this thesis and all the steps involved are presented.

4.1 Action research and TracIMo

The research method which is followed in this study is action research. As explained by Staron ([12], *"compared to other research methodologies, where the focus is either on the observation and learning or the evaluation, action research places more focus on the intervention (like in experiments), the context (like in case studies and observations), and learning"*. This is in line with the purpose of the study, where the research questions are to be answered by applying the intervention, TracIMo, on the context, 1928 Diagnostics. TracIMo is depicted in Figure 4.1 where the different boxes represent the steps that need to be carried out to design and deploy a traceability strategy.
TracIMo starts by data being collected and analyzed to understand the existing development process and derive *process goals* and *traceability goals*. Process goals state what the development process aims to achieve, while traceability goals, also known as *traceability usage goals* [11], only state what the trace links aim to achieve. These goals are used to produce a TIM in step 3 as well as analyzed by systematically assessing the process and traceability goals against each other in step 4. The TIM is also analyzed by systematically assessing it against the traceability goals in step 5. The first phase culminates in step 6 where the desired traceability process is derived. During step 7 the researcher will collect external data on available tools and then choose and customize the most suitable to accommodate the company's desired development process, TIM, and traceability process. If no available tool is suitable, in-house tools can be developed instead. In steps 8 and 9, the new process and tools are deployed and evaluated using a predefined measurement plan. The metrics should be designed to be able to evaluate whether the traceability goals are fulfilled. Step 10 is an important step for the long-term success of the traceability strategy where action is again taken to anchor the new process within the organization.
These steps can be compared to the steps required in an action research cycle: diagnosis, action planning, action taking, evaluation and learning, as seen in Figure 4.2 [12]. TracIMo's step 1 and 2 are the diagnosis, steps 3 to 6 are the action planning, 7 and 8 are the action taking and step 9 is the evaluation and learning.

Figure 4.1: Steps in TracIMo [2].



TracIMo is an iterative process where previous steps are revisited when needed, as shown by the dashed lines in Figure 4.1. This is similar to the cyclic nature of action research where all steps are in general repeated at least once [12].
It is also possible to introduce a traceability strategy bite-wise, i.e. not going for a holistic solution for the entire development process all at once, but instead focusing on a single development area at a time and then repeating TracIMo to cover more and more areas, which reflects the cyclic nature of action research as well. Due to the significant differences between different development areas, e.g. releasing new pipelines or making front-end improvements, we saw this approach as the most suitable for 1928 Diagnostics.

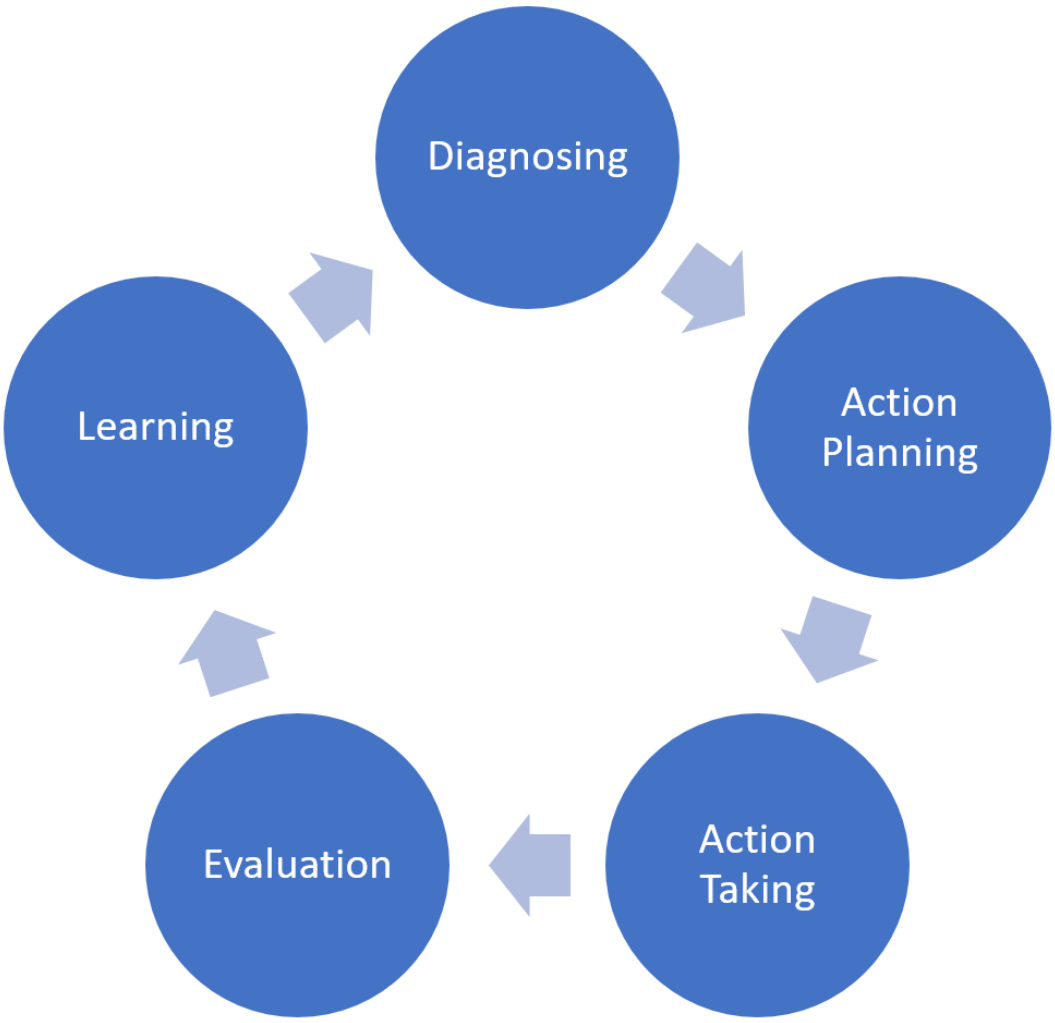
4.2 Implementation of TracIMo at 1928 Diagnostics

In this section, an overview of how TracIMo's steps were conducted at 1928 Diagnotics is given.

4.2.1 Steps 1 & 2 - Analysing Development Process and Traceability Goals

Following TracIMo, data collection started in steps 1 and 2, where interviews, which were recorded and transcribed, informal conversations, and observations took place. Originally the plan was to do this on site at the company, but due to the COVID-19 pandemic hitting Sweden only a few weeks into the study, nearly all data was collected remotely. The company managed to adapt to all developers working from home and observations to understand their development process could still be made

Figure 4.2: Simplified action research cycle [12].



during team meetings via Zoom. The researcher also got user accounts for digital tools that the company used, e.g. Slack, Google Drive and Jira, where more observations regarding the development process could be made. Interviews and informal conversations could also be conducted via Zoom and Slack without any issues. Little information was collected from procedure documentation reviews and inspection of requirement specifications due to the near lack of such artifacts.
QA/RA director and the chief technology officer CTO were interviewed individually two and three times respectively, while each software developer was interviewed once. In addition, one bioinformatician was interviewed and the results member checked with all four bioinformaticians. All bioinformaticians were not interviewed individually since the company wanted to minimize the time taken from them.
Questions asked to the QA/RA director were about his role at the company, traceability's role in different regulations and how traceability could benefit 1928 Diagnostics in general. TracIMo offers an interview guides for how to understand the development process and elicit traceability goals which the researcher used as input for preparing questions for semi-structured interviews with the CTO [1]. Most questions in this interview guide were used, although questions regarding how traceability is currently used were excluded due to the company's lack of experience using traceability. Many questions regarding how traceability could be beneficial in different development areas at the company were also added. In general, most questions asked to the CTO and developers were about the development process and different development areas. The developers were also asked how they thought traceability could benefit them.
Once the first round of interviews was completed, the data was processed by identifying and thematically coding quotes of interest from the transcriptions according to the topic of the quote. Codes used include *process goals*, *traceability goals* and different aspects of the development process such as *planning*, *documentation* and *testing*.
The first round of interviews revealed that different development areas have different processes. For this thesis, we narrowed our scope to focus on one development area after discussions with the CTO and QA/RA director.
After an area to focus on was decided upon, more interviews were conducted to get a deeper understanding of this area. Since this area related much more to the bioinformaticians than the software developers, it was primarily the same bioinformatician as in the first round of interviews who was interviewed. Once enough data was collected, a process model was created and traceability goals were specified in collaboration with the bioinformatician. These goals followed the standardised Goal/Question/Metrics (GQM) [14] format as recommended by TracIMo. According to this format, each goal has a purpose (e.g. "increase"), issue (e.g. "speed"), object (e.g. "of pipeline") and viewpoint (e.g. "for customers").
The GQM approach was also used to formulate questions and corresponding metrics for each traceability goal. These questions and metrics help to judge whether a goal has been achieved and measuring the success of the interventions. As recommended by TracIMo, scenarios for each traceability goal were also formulated. The scenarios are meant to be typical use cases where the trace links are used and helpful for evaluating the goals.

[1] `https://tinyurl.com/y3n96ldq`

Thereafter, the goals, questions, metrics and scenarios were sent to the CTO and QA/RA director for member checking. By now, the general process goals had also been refined to follow the GQM format.

4.2.2 Steps 3, 4, 5 & 6 - Deriving the TIM, Assessing the Goals and TIM against each others & Deriving the the Traceability Strategy

Using the data collected in steps 1 and 2 as input, a TIM was derived in collaboration with the bioinformatician, the CTO and another bioinformatician. The TIM was derived in iterations as more discussions were held and new issues were discovered. The traceability goals were also systematically assessed against the TIM in step 5, which led to one more iteration.
In step 4, the process goals and the traceability goals were systematically assessed against each others to make sure that the process goals needing traceability were properly supported by traceability goals and that no traceability goal was superfluous.
In the last step of the first phase of the TracIMo methodology, a suitable traceability process was derived in collaboration with the interviewed bioinformatician and another bioinformatician through various meetings.

4.2.3 Steps 7 & 8 - Selecting and customizing tool and Deploying process and tool

Discussions were held with the interviewed bioinformatician to decide which tools should be used and how they should be customized and combined to realize the entire traceability process. TracIMo has recommendations on how to select and customize external tools but these were not used since it became evident that proprietary tools would need to be developed. Whenever more input was needed, discussions with another bioinformatician were held. One of the software engineers also gave input on how a suitable architecture for the tools could be designed.
Since the chosen tools were based on existing code developed by employees at the company, it was most suitable for the interviewed bioinformatician to develop the tools. The researcher, the CTO and the rest of the development team were continuously updated on how the development was progressing during the daily stand-up meeting. Whenever issues in the planned changes were discovered, they could be discussed to find solutions for them. For example, an issue on how one of the trace links could be implemented was resolved.
Once a prototype of the tools was finished, they were tested by two bioinformaticians. Waiting until the tools were fully developed was considered but the prototype was seen as functional enough for the evaluation to proceed.

4.2.4 Step 9 - Evaluating process and tool

This step is to evaluate whether the implemented traceability solutions achieves the traceability goals and identify weaknesses. As suggested by the TracIMo, the trace-

ability goals and metrics from step 2 were taken as input for this step. Furthermore, long term evaluation of the strategy could not take place due to the time frame of this thesis.

4.2.5 Step 10 - Anchoring process and tool

Since anchoring the process and tools is something that naturally takes a long time and the entire traceability strategy was not fully implemented, it was not possible for the researcher to do this step. Therefore, this step is not covered in the results of this thesis.

5
Results

In this chapter, the results from implementing TracIMo at 1928 Diagnostics as described in the methodology will be presented. The results have been divided in sections according to TracIMo's steps, although some of the steps have been combined because certain steps were suitable to do in parallel.

5.1 Steps 1 & 2 - Analyze existing software development process and Identify traceability goals

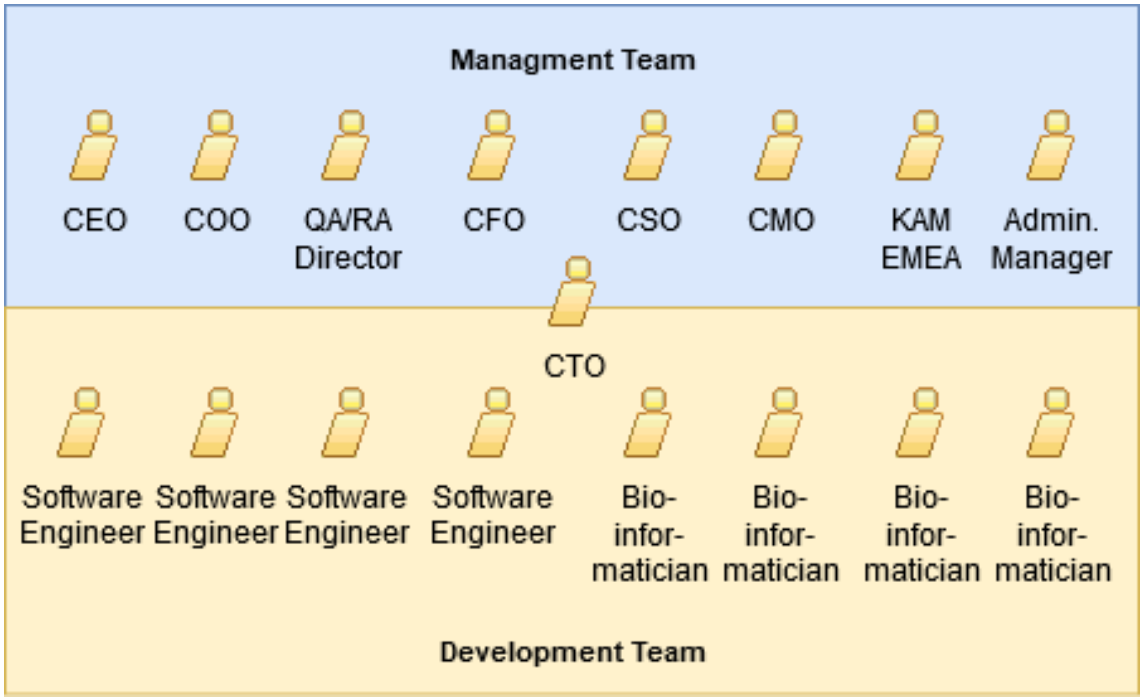
To complete the first steps of implementing a traceability strategy, data first had to be collected to understand the roles, development process and traceability goals. The CTO, QA/RA director, all software developers and one bioinformatician were thus interviewed at least once each.

5.1.1 Roles

The company is divided into a management team and a development team. In the management team there are two roles of interest for this study: the QA/RA director and the CTO. In this section the roles of the QA/RA director, CTO and development team will be discussed in detail. Other than these two, the management team also includes the Chief Executive Officer (CEO), Chief Operations Officer (COO), Chief Finance Officer (CFO), Chief Marketing Manager (CMO), Key Accounts Manager for Europe, Middle East and Africa (KAM EMEA) and Administration Manager. An overview of all roles that exist in the company can be seen in the Figure 5.1.

- **QA/RA director:** The QA/RA director's responsibility is to oversee aspects related to quality assurance and regulatory affairs. The director mostly engages with the development team whenever complying to a regulation becomes relevant, which isn't that often. As such, the director's role is more on a strategic level, working out how the company should position itself in relation to different regulations from a long-term perspective. Included here is collaborating with Chalmers to do research on how aspects of the development process can be refined.
- **CTO:** The CTO, who is also the product owner, has a central role in the development of the platform and is therefore part of the development team as well as the management team. Albeit in collaboration with the rest of the management team, he is the main driver of the strategic work behind creating the product development road map. By talking to customers and reading

Figure 5.1: Overview of the company, divided between a management and development team.



various articles, his objective is to understand the market and analyze new product development possibilities. He then communicates to the development team which strategic areas and end goals for the next six to twelve months of development are. Through planning sessions, they subsequently form project groups for the strategic areas and together decide where to start working. Depending on the size and scope, developers and the CTO himself can spend anywhere between weeks and several months on these projects. Throughout development, the CTO has regular discussions with the developers to support them in any way needed.

- **Development team:** The development team consists of eight developers and is led by the CTO. The eight developers are equally divided between two distinct roles: software developers and bioinformaticians. Software developers are responsible for developing the platform's front-end as well as large parts of the back-end including its architecture. Bioinformaticians on the other hand are responsible for developing and validating another part of the back-end, namely the various pipelines through which samples are analyzed.

5.1.2 Development process

The development team follows a general development process followed by all developers which is further refined depending on the development area. For example, there is specific development process that a bioinformatician goes through to develop and validate new pipelines. This development process also varies depending on the kind of pipeline.

5.1.3 General development process

1928 Diagnostic development process can in general be described as one that embraces the agile methodology. Core to the methodology is the agile manifesto and the twelve principles listed below [17]. In this section, a summary of 1928 Diagnostics' general development process and how it relates to some of the Agile Manifesto's principles is given.

"Individuals and interactions over processes and tools.
Working software over comprehensive documentation.
Customer collaboration over contract negotiation.
Responding to change over following a plan." [17]

The twelve principles:

1. "Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity–the art of maximizing the amount of work not done–is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly." [17]

- **Release schedule:** As stated in the first and third principles, continuous delivery of features valued by customers is really what the development process is primarily focused on. The development team works in three week long sprints but succeeds in delivering many new releases throughout the sprints thanks to an emphasis on continuous integration and continuous delivery (CI/CD). CI/CD is the practice of having regular releases of small updates, rather than few releases of big updates, by working in smaller batches. As explained by the CTO, this also minimizes the need for coordination and change impact analyses, since each release is so small. However, for CI/CD to be possible, it is important for the master branch to always/nearly always be in a deployable state and that there is a fast and reliable test suite, or there will be too many things to fix and worry about for developers to make regular releases.

- **Requirements:** Moving on to being able to welcome changing requirements, this is hardly a problem at all since requirements are generally not formally documented. Unlike many other agile companies, 1928 Diagnostics does not have a business analyst whose responsibility it is to create and delegate detailed tickets containing user stories, tasks or requirements for the developers to work on. They believe a free flow of mental requirements, i.e. that aren't formally documented but instead briefly discussed during meetings or through different tools such as Slack, suits them better. This allows them to make changes in direction with very little overhead, for example having to update tickets.

  They see the best way to enforce that a requirement, regardless if it has been formally specified, is being fulfilled is with tests. As stated in the agile manifesto, working code is more important than comprehensive documentation. As with any other piece of documentation or artifact, formal requirement specifications are thus only created when there is a clear value, not just when there is a small probability that it will be good to have in the future. Ideally, developers should do a cost-benefit analysis to evaluate if creating an artifact is worth creating. In general, development is thus not reliant on documentations/requirement specifications.

  This can also be seen as a weakness though, since a robust process for managing requirements is beneficial from a regulatory perspective. There have been attempts to introduce a process for having requirement specifications in their backlog, but these were unsuccessful. Thus, the processes for managing external requirements have until now been very lightweight. For example, what the company did when HIPAA requirements had to be fulfilled, was to simply take an Excel sheet containing the requirements and check them off one by one. It is not often that external requirements have to be fulfilled though, hence no mature process for managing specified requirements has been anchored.
- **Sprint meetings:** The team has three different sprint meetings during each sprint. Once a sprint has come to its end, a sprint demo, where features that have been released to the product are demonstrated, is held.

  Thereafter a sprint refinement meeting is held, where suitable sprint goals for the coming sprint are discussed and prioritized. Sprint goals can be thought of as milestones within a project for developers to focus on achieving during the coming sprint. The development team tries to make sure that their sprint plans are expressed as goals rather than tasks, partly since they want to avoid heavy planning/monitoring processes. Sprint goals are mainly based on the product development road-map, which the CTO has in the back of his mind, and the developers' own opinions on how to slice projects into various sprint goals. The sprint refinement meetings is also an occasion where the team from time to time reflects on how to make their development process more effective, as suggested by the twelfth principle.

  A new sprint is thereafter kicked off with a sprint planning meeting, where the sprint's goals are briefly discussed and finally determined.
- **Development:** In accordance with the fifth and eleventh principles, the team then puts faith on the developers to be motivated and able to organize, plan and carry out the development themselves. It is thus important that the de-

velopment team consists of small, self-organizing, autonomous units, rather than relying on top-down micromanagement. Relying on developers' ability to be independent does not mean that they are not supported though, which is also part of the fifth principle. As per the fourth principle, daily stand up meetings, which are attended by all developers as well as the CTO, are held. These meetings offers them the platform to share how things are progressing and get support whenever needed. In addition, weekly meetings, one for bioinformaticians and another for software developers, are held where developers have the chance to discuss subjects in more detail. During these meetings, time is also spent to reflect on how to make their development process more effective, as described in the twelfth principle.

- **Tool support:** Although the company does not rely on tickets for planning and driving development, the company still finds some use of tools such as the issue tracking product Jira [18]. The extent to which they have used Jira has fluctuated through time as they occasionally decide to use more or less Jira features in their pursuit towards a development process that the entire team feels content with. As professed in the tenth principle though, simplicity is a ubiquitous concern when it comes to their development process. As such, they generally avoid changes that would require more time being spent on maintaining a tidy backlog of tickets. Tickets that are created thus tend to be sparse on details. At the time that this study started, the main use of Jira was for bug tracking, although they were making efforts to improve git branch management by leveraging its integration with the version control system Bitbucket [19], where their source-code is hosted. Work that does not produce code, i.e. that does not affect the source-code, is generally not documented in tickets or any form other than whatever artifacts are affected by the work itself. Furthermore, creating tickets for work that does produce code is only encouraged but not enforced, meaning that Jira cannot give a full picture of what code changes have been done.

  As for keeping track of current sprint goals, when this study began only a physical whiteboard was used. However, a couple weeks into the study all employees started working from home due to the COVID-19 pandemic, meaning that the whiteboard could no longer be used. Therefore, the development team then began documenting their sprint goals digitally in Google Drive by creating documents for each sprint meeting containing meeting minutes and sprint goals. In addition, Google Drive is sometimes used for taking shared notes related to a project multiple developers are involved in. For example, developers might create a document containing a description of the project, the goals and updates as the project progresses. However, this practice is not standardized or enforced, developers are free to do however they please. The team also has also used the project management tool Basecamp [20] for the same purpose but as of late been using it less and less.

  Another important tool they use is Amazon Simple Storage Service (S3) [23] provided by Amazon Web Services (AWS) [24], which is used for data storage. For example, large files that aren't needed by the platform are stored here.

- **Communication:** 1928 Diagnostics values face-to-face conversations in favor

of documentation, which goes in line with the sixth principle. Whenever possible, time spent on maintaining thorough documentation and detailed tickets is minimized and letting discussions be the development process' driving force. Obviously this is not to say that all conversations are face-to-face though. Proof of this their extensive use of the communication platform Slack in which many announcements, discussions and casual conversations are hosted. Also worth mentioning is that since employees started working remotely due to COVID-19, all meetings have been held on the video communication application Zoom instead of actually face-to-face.

- **Team spirit:** Finally, an important ceremony to promote a sustainable development as suggested in the twelfth principle, is the team retrospective where all employees gather for expressing reflections once a month. Here, everyone shares the positive and negative experiences they have had during the month and the team discusses how practices and routines can be changed to maintain team spirit and foster a better workplace.

Process models describing the development process in general are shown in Figures 5.2 and 5.3.

5.1.4 General process goals

Based on the information collected from the CTO and QA/RA director, the following process goals for the development process in general were identified. To make sure that these goals were accurate, member checks with the CTO and QA/RA director were conducted. Traceability goals for the general development process were not documented, since these varied between different areas of development.

- **CI/CD**
  - **Goal 1:** Maintain/improve development process's CI/CD capabilities from the development team's point of view.
  - **Goal 2:** Increase how often master branch is deployable from a developer's point of view.
  - **Goal 3:** Improve the reliability of the test suite, e.g. by increasing test coverage, from a developer's point of view.
  - **Goal 4:** Increase the speed of the test suite from a developer's point of view.
- **Requirements and Documentation**
  - **Goal 5:** Maximize cost-benefit of created artifacts by avoiding to create requirements and documentation (or other artifacts) that don't have a positive cost-benefit from the from the development team or customers' point of view.
  - **Goal 6:** Decrease risk of requirements (including internal requirements that have not been documented) not being fulfilled by creating enough tests from the development team's point of view.
  - **Goal 7:** Improve accessibility of requirements, documentation, tests or other artifacts by storing them close to the related code implementation within the code-base if possible from a developer's point of view.

Figure 5.2: Simplified process model for development process in general. Rectangles represent activities while arrows show the input and output of the activities. Activities in the yellow area are done by the development team while activities in the blue area are done by the management team. The CTO is part of both teams and thus included in both areas.

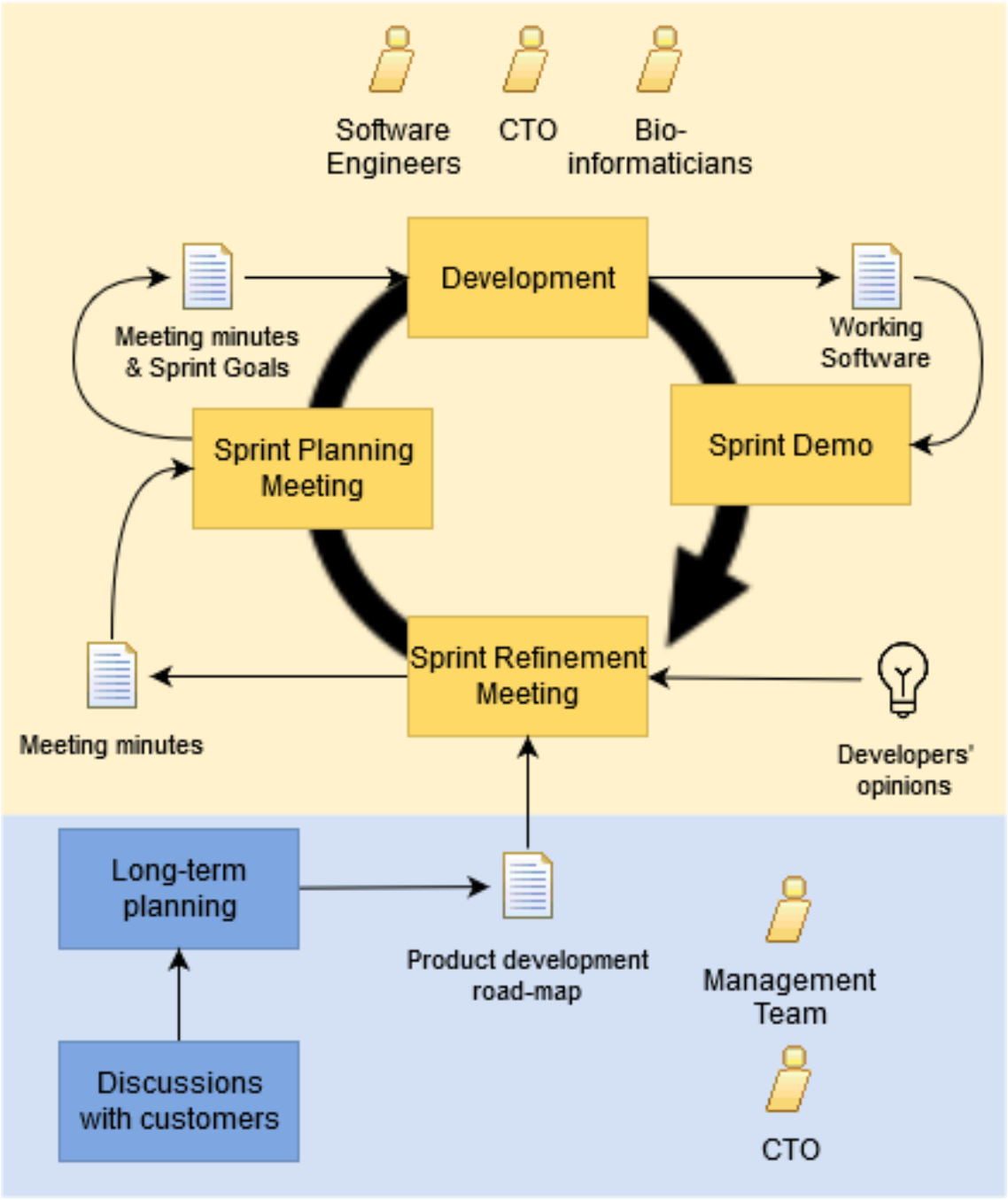
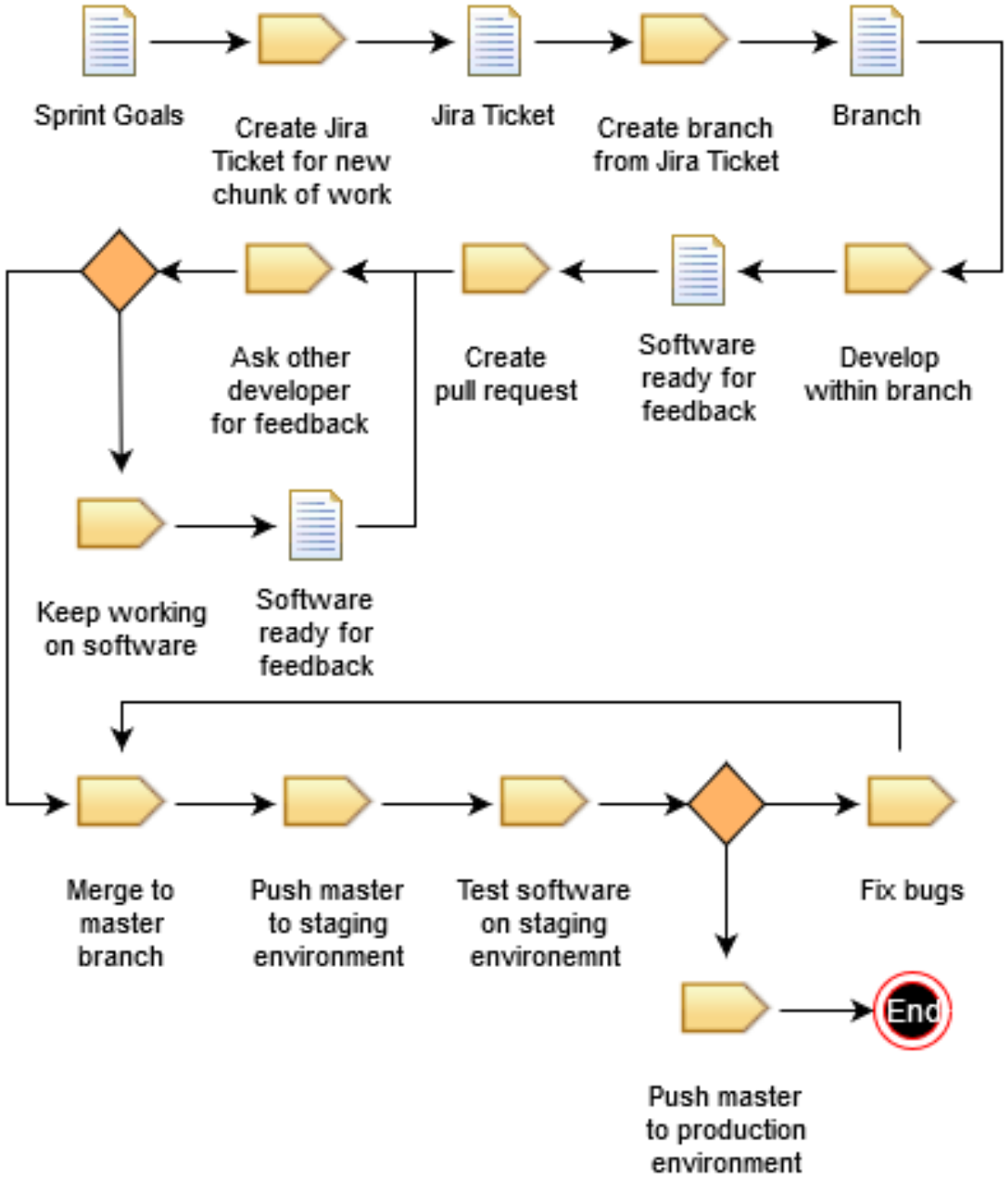


Figure 5.3: Process model for what "Development" in Figure 5.2 typically looks like. The development process followed by a bioinformatician to develop and validate a new pipeline is very different. Pentagons represent tasks while the diamonds represent decision points.



  - **Goal 8:** Improve process for managing documented requirements from the development team's point of view.
- **Planning**
  - **Goal 9:** Maintain development team's level of self-organization from the development team's point of view.
  - **Goal 10:** Minimize top-down micro-management from the CTO's point of view.
- **Regulations and external requirements**
  - **Goal 11:** Implement formalized process for fulfilling and maintaining compliance to currently relevant external regulations and external requirements (e.g. HIPAA) from the QA/RA director and the development team's point of view.
  - **Goal 12:** Maximize benefit of new formalized processes by taking into account how they could be used in the future for complying to medical device regulations from the QA/RA director and the development team's point of view..

5.1.5 Determining which area to introduce a traceability strategy for

Different development areas within the company have their own development processes. Since each development process deals with different kinds of artifacts, it was not feasible to introduce an all-encompassing traceability strategy that works with all development processes and artifacts during this study. As such, we had to identify one development process where introducing a traceability would be suitable and of high value for the company.

5.1.5.1 Traceability strategy for the development process in general

The first option was to implement a traceability strategy involving artifacts used in the development process in general, i.e. artifacts that are used in a wide range of development areas. These artifacts could for example be artifacts used for planning such as tickets and documents.
Throughout the interviews conducted, minor issues relating to such artifacts and which potentially could be solved with traceability were identified. For example, there were no trace links between documents in Google Drive containing information about sprint goals or projects and associated tickets in Jira, which might explain why such documents sometimes got buried and forgotten. However, after having discussed with the CTO and QA/RA director, addressing these issues were seen as low-priority, which in part can be explained by the process goal of avoiding heavy planning/monitoring processes.
Since the CTO was mostly familiar with use of traceability in relation to external requirements, he was presented with the traceability benefits listed in the background of this thesis. The idea behind this was to make the CTO aware of other applications that could potentially be useful for the development team. The reason why the CTO was seen as the most suitable to do this kind of interview with was

because he is knowledgeable in all development areas and because changes to the development process would need his approval and support.
Although discussing these benefits was helpful to get a deeper understanding of how the development team operates, this approach was unsuccessful in eliciting suitable traceability goals for the sake of this thesis. Product conformance, project accountability and organizational learning were not of much interest since the general development process normally didn't specify requirements or produce documentation. By striving for CI/CD and a reliable test suite, the CTO also didn't see a need for change impact analyses before doing changes. Although process conformance and baseline reproducibility were seen as important, concrete ways that traceability could be introduced for these benefits were not identified.

5.1.5.2 Traceability strategy for HIPAA requirements

The second option, which had been discussed with the QA/RA director, was to introduce a strategy for implementing traceability between the already fulfilled list of HIPAA requirements and verification, i.e. tests for each requirement. As explained by the QA/RA director, the reason why traceability between requirements and verification, as opposed to implementation, is of interest, is due to regulatory and practical reasons. Regulations care about there being adequate tests ensuring the desired performance, how these tests are then passed is of less interest. Furthermore, a developer will often be able to locate the code fulfilling a test without additional trace links by inspecting the test itself. Implementing this strategy was seen as much more beneficial than the fore-mentioned strategy since continued HIPAA compliance is more important and a similar strategy could probably be used later when medical regulations become relevant. However, implementing such a strategy also has its challenges in the form of two things that are missing. Firstly, a formalized requirement management system for such requirements would need to be in place, since only listing the requirements in an Excel sheet, where the HIPAA requirements currently are, is insufficient. The reason for this is because there are no convenient tools for creating trace links between cells in an Excel sheet and files in a git repository. This is not to say the entire development process would need to be overhauled, only the process whenever requirements stemming from regulations are in play. As stated by the CTO, separate development process with/without regulations is preferable since they want to avoid heavy processes whenever it is not needed. Secondly, verification would need to be added for at least some of requirements. Although the requirements have been fulfilled, there are currently no tests for them. Thus, a considerable amount of efforts would need to be invested before traceability could be introduced in this area.

5.1.5.3 Traceability strategy for validation of pipelines

During an interview with one of the software developers, the developer mentioned the development process bioinformaticians do for creating and validating new pipelines as an area where traceability perhaps could be beneficial.
To understand more about this complicated development process, a bioinformatician was interviewed three times. As explained by the interviewed bioinformatician, a lot

of their time goes into validating that the pipelines for analyzing samples perform well. Although research tools do not have the strict regulations medical devices do, it is still important that the customers can rely on the company's products. There are many different pipelines and components needed by the pipelines being developed for different kinds of analysis, each of which has slightly different development and validation processes.

One component necessary for outbreak tracing of a certain pathogen is having a reliable *cgMLST schema* for that pathogen, which the bioinformaticians generate themselves. Although the cgMLST schemas are kept secret, it happens that customers ask for validation documents explaining how a cgMLST schema was generated and validated, which is why the company started creating one such document for each released cgMLST schema. In these documents, the bioinformaticians refer to various artifacts that went into the generation and validation of the cgMLST schema. However, a formalized process for managing these artifacts is currently lacking, which sometimes causes them to get lost and the bioinformaticians needing to put time and effort in reproducing the artifacts. Furthermore, little effort have been put into establishing traceability between these artifacts even though it could help them keep track of the artifacts. The results of these interviews were member checked with the remaining bioinformaticians to ensure that they agreed with the diagnosis of the problem.

After having talked to the CTO and QA/RA director, it was decided that introducing a traceability strategy in this area was the most suitable for various reasons. First of all, it would be very beneficial for the generation and validation of cgMLST schemas without having to put other things in place first. Second, as stated by the QA/RA director, a similar strategy could be beneficial when dealing with medical device regulations. Just like with medical devices development, this area has the need for documentation explaining how something was created and validated, unlike the HIPAA requirements. Although there are no external requirements in this context, there still are internal requirements regarding the performance of a cgMLST schema. These requirements are only mentioned in the validation documents implicitly, but if a process for managing these requirements is added in the future, the process could be used in combination with external requirements.

5.1.6 Development process for generation and validation of cgMLST schemas

The bioinformatician was interviewed one more time to understand more about what a cgMLST schema is and the artifacts and development process used to generate and validate one. The bioinformatician later also had a key role in the development of the traceability strategy and will henceforth be referred to as the interviewed bioinformatician.

Below, an explanation of the development process is given by going through the involved artifacts one by one. In Table 5.1, an overview of where the artifacts are currently saved is presented. In Figure 5.3, a process model describing the development process for generating and validating a new cgMLST schema are shown.

1. **cgMLST schema:** A cgMLST schema, short for core genome multilocus se-

quence typing schema, is a collection of *core genes* that one can expect to be present in most samples of a certain species, or in this case in most samples of a particular pathogen such as the Mycobacterium tuberculosis bacteria which causes Tuberculosis. CgMLST schemas are used as a reference point while analyzing and comparing sample sets and it is thus important that schemas are representative for the species and can handle the species' diversity. Bioinformaticians often need to generate and try several versions of cgMLST schema until a well-performing candidate is finally released.

Once a schema is to be released, a copy is uploaded to the main repository in Bitbucket where all the products are. Under-performing unreleased schemas, which sometimes can still be relevant to mention in the validation document to motivate how the final schema was generated, do not have a designated location. Sometimes they are saved in the S3 storage while other times they are saved locally where they can get lost. Unreleased schemas should not be uploaded to the main repository though since schemas are quite bulky and the development wants to prevent the main repository from getting bloated by keeping bulky artifacts not needed by any product in other locations.

It is also possible for a second version of cgMLST schema to be released and replace the first released version. However, this only happens if it is later discovered that the first version doesn't perform well enough. This is due the fact that replacing a released cgMLST schema requires customers to be notified and their previously analyzed samples to be analyzed again using the new version. If this happens, the first version is to be left in the main repository to make it clear that multiple versions have been released.

2. **Reference genomes:** A species' cgMLST schema is generated using a collection of genomes belonging to the species, referred to as reference genomes. A bioinformatician thus starts by searching for reference genomes from a public server hosted by NCBI RefSeq [25], a public database containing high quality genome sequences, and selects one as a *seed genome*, which plays a central role. However, it sometimes happens that certain genomes in NCBI RefSeq are mislabeled, in which case a bioinformatician will exclude these from the reference genomes.

   Since reference genomes are easy to fetch from NCBI RefSeq whenever needed using their unique IDs, they are not downloaded and stored anywhere else.

3. **Pipeline for generating cgMLST schemas:** Next, a bioinformatician feeds the reference genomes and selected seed genome to a specific pipeline the company has created for generating candidate cgMLST schemas. All cgMLST schemas are generated using the same pipeline regardless of species, but for each species there will be an optimal set of parameters to run the pipeline on. However, a bioinformatician can only guess which parameters to use based on articles they have read and must later try different parameters until a suitable cgMLST schema is produced. Later knowing which parameters were used for a certain cgMLST schema relies on a bioinformatician taking proper notes. It is also possible for the pipeline itself to get updated, but this happens very seldom.

   The pipeline is stored in Bitbucket although not in the main repository since

no product needs to access it and it is seen as bulky. Instead, it is stored in a sandbox-repository.

4. **Datasets:** Before a candidate cgMLST schema is released, 1928 Diagnostics requires that the schema is benchmarked against one or several collections of samples of the species, referred to as datasets. In other words, each dataset contains samples from the pathogen that the cgMLST schema is intended for. This is to make sure that the cgMLST schema contains the right genes and covers the species' diversity. Thus, a bioinformatician must next search for suitable datasets. Most datasets come from publicly available research results, although a few of them may also come from 1928 Diangostics' collaborators. Additional information about the datasets such as the number of samples, hospitals and countries are also often included in the validation document, although this is not strictly standardized.

   Before benchmarking, the datasets should also ideally be properly implemented within the platform's code base, meaning that samples are properly stored and indexed in a dedicated datasets-repository where the main repository can easily access them. Alternatively, the samples can be retrieved directly from the source's public server each time they are to be used. However, this has the drawback that benchmarking then takes more time since the samples, which are quite large, have to be downloaded each time. Unfortunately, the process for properly implementing is rather complicated and has not been sufficiently prioritized by the development team, leading to many datasets never being implemented properly.
5. **cgMLST analysis pipeline:** Thereafter, to evaluate the performance of the cgMLST schema, each sample included in the dataset(s) must be run through a pipeline, here referred to as the cgMLST analysis pipeline and not to be confused with the pipeline for generating a cgMLST schema. What this pipeline does is to compare the schema with the samples and calculate the percentage of core genes, i.e. genes in the cgMLST schema, included in the sample, referred to as the *fraction of core*. A good cgMLST schema will have a high fraction of core. The same cgMLST analysis pipeline is used for all pathogens, except for that different parameters are used. To facilitate the process of running each sample through the cgMLST analysis pipeline and later evaluating the candidate cgMLST schema, a benchmarking script is used.

   The cgMLST analysis pipeline is saved in Bitbucket in the main repository together with the rest of the platform. Note that the cgMLST analysis pipeline is not the only analysis pipelines, there are others that evaluate other things.
6. **Benchmarking script:** A benchmarking script is a script that automatically feeds all samples in one or several dataset to one or several analysis pipelines, in this case to the cgMLST analysis pipeline, and generates a benchmark result for each sample.

   First, the benchmarking script must access the dataset. To do this, a bioinformatician must hard code into the script which dataset to use. If the dataset has been properly implemented, this is pretty straight forward. However, since the datasets often aren't, a bioinformatician will normally have to tell the script to retrieve the dataset directly from the source's public server.

Next, a bioinformatician must hard code which cgMLST analysis pipeline, including which parameters, and which cgMLST schema to run.

All this hard coding leads to many benchmarking scripts being created and tweaked. Currently, these scripts are mostly stored locally by the bioinformatician who created them where they can get lost. The company has thus thought about creating a single benchmarking script where a bioinformatician can specify the input without any hard coding. Such a script was once created but has not been used for a long time and therefore out-dated. One challenge in realizing this is that for a dataset to be used, it would always need to be properly implemented.

7. **Benchmark results:** The benchmark result for each sample are at the end combined and compressed into a tarball-file by the benchmarking script.

   For a schema to be released to the product, 1928 Diagnostic has an internal requirement saying that all samples need to have a fraction of core higher than 95%, i.e. that more than 95% of core genes are included in each sample. However, upon further inspection, samples with a lower fraction of core may be ignored if they did not have sufficient sequencing coverage, meaning that a sample was not sequenced thoroughly enough and is more likely to contain errors. If a schema performs poorly, it may be a sign that the wrong parameters are being used or that the reference genomes do not cover the species' diversity and that more reference genomes must be found.

   Benchmark results are sometimes but not always uploaded to the S3 storage. As a consequence, benchmark results relevant for validating a cgMLST schema sometimes get lost, which can cause problems if the schema later needs to be re-validated. Benchmark results are also quite bulky and not needed by any product though so they should not be uploaded to the main repository.

8. **Validation document:** Once a well performing cgMLST schema is generated, the schema is finally released to the platform, allowing the customer to do outbreak tracing of the pathogen. It is now time for a bioinformatician to create a validation document containing the information needed to prove that the schema has been validated, i.e. benchmarked properly.

   Although the contents of a cgMLST schema's validation documents varies from schema to schema, there are standardized parts that all validation documents should include. The first standardized part is an overview of how the cgMLST schema was created, including which seed genome, reference genomes and parameters in the pipeline for generating cgMLST schemas were used. The second is an overview of how the cgMLST was benchmarked, including which dataset(s) and a summary of the benchmark results (e.g. the median and mean fraction of core). On top of these standardized parts, a bioinformatician will sometimes include more specific details on the creation process of the cgMLST schema.

   The validation documents are currently created and stored in Google Drive.

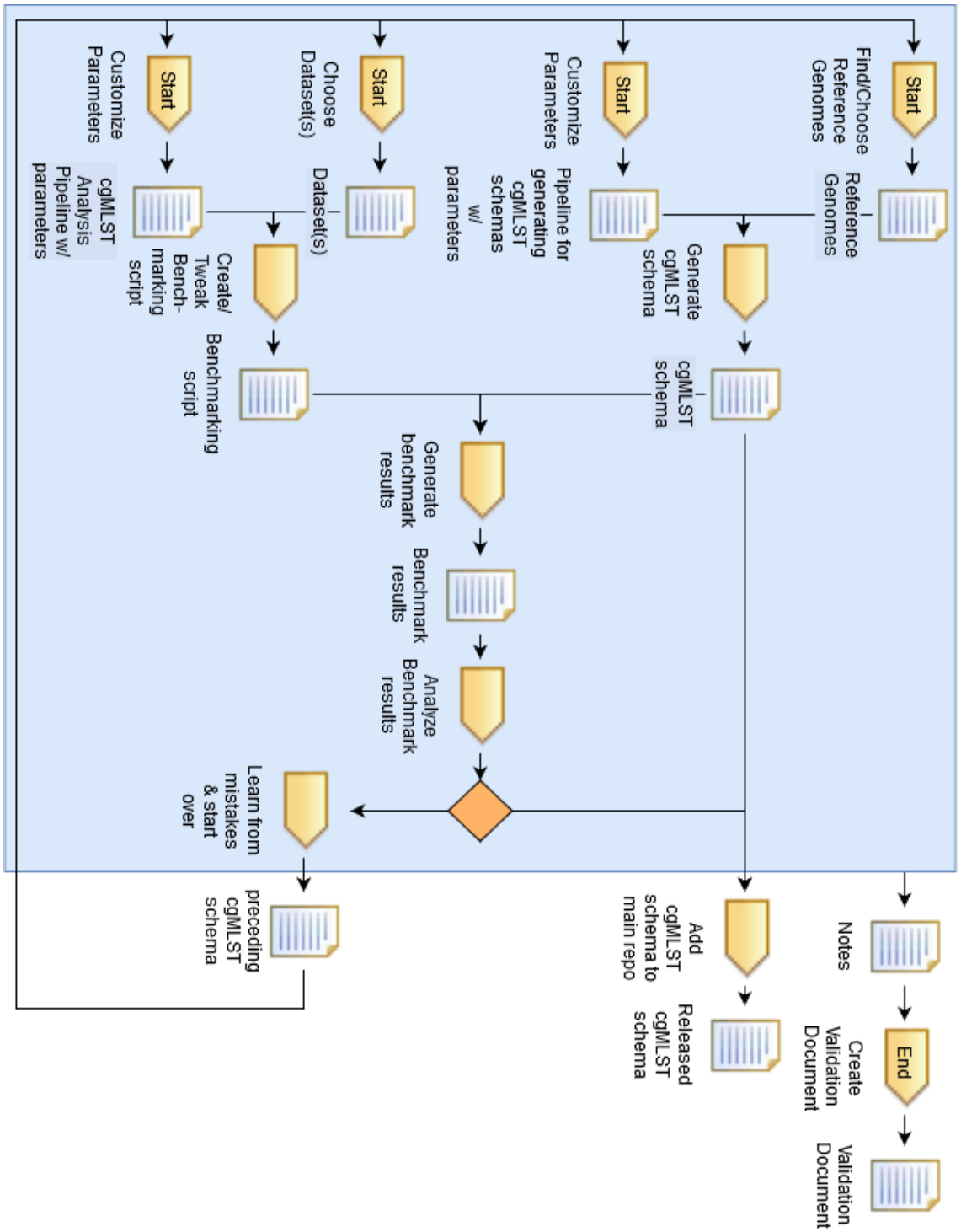
9. **Notes:** Throughout the development process, a bioinformatician will take notes of the entire process to remember important details and facilitate the creation of the validation document. For example, a bioinformatician will often take notes of the the benchmark results to avoid having to look for the

results later when creating the validation document. The notes are typically saved locally.

Table 5.1: Overview of where artifacts are currently saved.

	Main repo	Datasets repo	Sandbox repo	Public server	S3	Locally saved	Google Drive
Released cgMLST schema	Here						
Unreleased cgMLST schema					Sometimes here	Sometimes here	
Reference genomes				Here			
Pipeline for generating cgMLST schemas			Here				
Datasets		Sometimes here		Sometimes here			
cgMLST analysis pipeline	Here						
Benchmarking script	Used to be here				Sometimes here	Mostly here	
Benchmark results					Sometimes here	Mostly here	
Validation document							Here
Notes						Here	

Figure 5.4: Process model for generation and validation of cgMLST schemas. Tasks and artifacts in the blue area are input for the created notes.



5.1.7 Process goals for creation and validation of cgMLST schemas

Similarly to how the general process goals were elicited, process goals for this development area were identified based on the discussions with the interviewed bioinformatician, CTO and QA/RA director and listed in Table 5.2. These were then also member checked by the CTO, interviewed bioinformatician and QA/RA director.

Table 5.2: Process Goals.

Goal 1:	Improve process for creating and validating cgMLST schemas from a bioinformatician's point of view by having all artifacts necessary to reproduce and verify the contents of a validation document uploaded so they don't get lost.
Further details:	This also requires artifacts that are subject to changes to be version controlled, otherwise only the latest versions of an artifact will be accessible.
Goal 2:	Improve standardization and quality of validation documents and facilitate the creation of them from a bioinformatician's point of view.
Further details:	Since validation documents are created manually, it is possible for bioinformaticians to make errors while copying information from relevant artifacts or forgetting to add parts all together. Furthermore, there is no template or standardized process for creating them, meaning that there are unmotivated differences between them. Some validation documents are even missing the standardized parts that all documents should include.
Goal 3:	Improve change control of validation documents.
Further details:	As explained by the QA/RA director, change control is important from a regulatory perspective. Change control means that if an change is to be done, one should have a systematic process for handling artifacts affected by this change to ensure that new faults aren't introduced. As for change control of a validation document, one possible scenario is when a new dataset relevant for the pathogen in question is discovered. This could lead to new benchmark results and the validation document needing to be updated. However, if a released cgMLST is replaced, then a completely new validation document would be created, meaning that change control of validation documents doesn't have to consider this scenario. Note that a released cgMLST schema is only replaced if it is discovered that it does not perform well, hence change control of cgMLST schemas is not relevant. In other words, a released cgMLST schema will never be changed just because related artifacts are changed.
Goal 4:	Maintain lightness of main repository by keeping out bulky artifacts that are not needed by any product from the development team's point of view.

Table 5.2: Process Goals. *Continuation.*

Further details:	Such artifacts include benchmark results, unreleased cgMLST schemas and the pipeline for generating cgMLST schemas. This is to avoid the main repository getting bloated and leading to slower development.
Goal 5:	Reduce number of benchmarking scripts used to facilitate maintenance and continued improvement of benchmarking capabilities from a bioinformatician's point of view. When possible, existing scripts should be built upon instead of creating new scripts.
Goal 6:	Increase number of datasets that are properly implemented so that benchmarking takes less time from a bioinformatician's point of view.

5.1.8 Traceability goals and measurement plan for creation and validation of cgMLST schemas

In parallel with the identification of process goals for this development area, traceability goals were also identified and have been listed in Table 5.3 along with rationales explaining why each goal is relevant. Questions, metrics and scenarios were then derived for each goal, as described in the methodology.
However, it is not only the trace links' added benefit that is important to judge the success of a traceability goal, but also the cost, i.e. effort, of using the process and tools to create and maintain the trace link. Ideally, the combined cost of creating and maintaining trace links and then performing tasks *with* the trace links, should be lower than the cost of performing the same task *without* any trace links [4]. It was thus decided that each question should be answered with two different kinds of metrics: benefit and cost metrics. Benefit and cost metrics in Table 5.3 have been marked with a (B) and (C) respectively. Note that certain traceability goals require the same trace links as other traceability goals. Therefore, to avoid repetition, certain questions don't have a cost metrics (see notes).
Next, a measurement plan for the gathering data on the metrics was made. Since all traceability goals aim to improve something from a bioinformatician's point of view, data for all metrics were to be collected during a group interview with two bioinformaticians. The reason why only two of the bioinformaticians were to be interviewed was because the other bioinformaticians have less experience creating cgMLST schemas.
However, since getting feedback on the traceability strategy from the CTO and QA/RA director was important for them to evaluate how to work with traceability moving forward, not just in the context of cgMLST schemas, a focus group with them and both bioinformaticians was held afterwards. This focus group would aimed to keep the questions and discussions on a higher level than the group interview, meaning that not necessarily each traceability goal and metric would be covered individually.
The questions used in the group interview and focus group were only finalized after step 8, i.e. once the traceability strategy had been deployed in order for the questions to really fit.

Table 5.3: Traceability Goals, questions and metrics.

Goal 1:	Improve accessibility from a validation document to the exact cgMLST schema, benchmark results and script for generating validation documents *(see Goal 4 for explanation of this script)* that were used to generate the validation document. This does not only apply to the latest released version of a cgMLST schemas, but also to older released versions.
Rationale	This allows for the content of any validation document to be verifiable and reproducible. Other artifacts that are referred to in the document are to be accessed via the fore-mentioned artifacts, hence not mentioned in this goal. Fulfilling this goal for older released cgMLST schemas is important because a customer might still want to validate older results that they produced with such a cgMLST schema.
Question 1:	Is it easier to access the mentioned artifacts?
Metric 1.1:	(B) Opinions and perceptions on the accessibility from a validation document to the mentioned artifacts, with and without trace links.
Metric 1.2:	(C) Opinions and perceptions on the cost of using the process and tools for creating and maintaining the trace links from a validation document to the mentioned artifacts.
Scenario 1.1:	Given the latest version of a validation document, it should now be easier to access the artifacts.
Scenario 1.2:	Given an older version of a validation document, it should now be easier to access the artifacts.
Goal 2:	Improve accessibility from a cgMLST schema to the exact versions of artifacts that went into the generation of the schema from a bioinformatician's point of view, i.e. pipeline for generating cgMLST schema (including parameters used), reference genomes and preceding cgMLST schemas. This does not only apply to the latest released version of a cgMLST schema, but also to older released/unreleased versions.
Rationale:	This allows for any cgMLST schema to be reproducible and one can verify that a cgMLST schema was in fact created as stated in the validation document. Fulfilling this goal for older released versions is important because a customer might still want to validate their old results that were produced with a previously released cgMLST schema. Although not as important, improved accessibility to preceding unreleased cgMLST schemas (and the artifacts that went into generating them) is still relevant because there are cases where decisions are explained in the validation document by referring to failed attempts.
Question 2:	Is it easier to access the mentioned artifacts?

Table 5.3: Traceability Goals, questions and metrics. *(Continuation)*

Metric 2.1:	(B) Opinions and perceptions on the accessibility from a cgMLST schema to the mentioned artifacts, with and without trace links.
Metric 2.2:	(C) Opinions and perceptions on the cost of using the process and tools for creating and maintaining trace links from a cgMLST schema to the mentioned artifacts.
Scenario 2.1:	Given the latest released cgMLST schema, it should be easier to access the artifacts.
Scenario 2.2:	Given an older released cgMLST schema, it should be easier to access the artifacts.
Scenario 2.3:	Given a preceding unreleased cgMLST schema, it should be easier to access the artifacts.
Goal 3:	Improve accessibility from benchmark results to the exact versions of artifacts used to produce the benchmark results from a bioinformatician's point of view, i.e. the cgMLST schema, dataset, benchmarking script and cgMLST analysis pipeline (including parameters used). This does not only apply to benchmark results of the latest released version of a cgMLST schema, but also for older released/unreleased versions.
Rationale:	This allows for any benchmark result to be reproducible and verify that the results stated in the validation document are correct. Improved accessibility in the opposite direction is not seen as necessary. Upholding this goal for older versions is important because a customer might still want to validate old results produced with a previously released cgMLST schema.
Question 3:	Is it easier to access the mentioned artifacts?
Metric 3.1:	(B) Opinions and perceptions on the accessibility from benchmark results to the mentioned artifacts with and without trace links.
Metric 3.2:	(C) Opinions and perceptions on the cost of using the process and tools for creating and maintaining trace links from benchmark results to the mentioned artifacts.
Scenario 3:	Given some benchmark results, it should be easier to all of the artifacts.
Goal 4:	Improve creation process of validation documents from a bioinformaticians point of view by allowing standardized parts to be automatically generated with the help of a script and trace links to related artifacts.
Rationale:	Using a script for generating validation documents based on related artifacts (that are located with previously created trace links) would support improving the standardization and quality of validation documents as well as facilitating the creation of them. The reason why only standardized parts of the validation document can be automatically generated is because the other parts are more unique and have to be done manually.

Table 5.3: Traceability Goals, questions and metrics. *(Continuation)*

Question 4.1:	Are the validation documents of higher quality?
Metric 4.1:	(B) Opinions and perceptions on the quality of validation documents, with and without trace links and the script.
Question 4.2:	Are the validation documents more standardized?
Metric 4.2:	(B) Opinions and perceptions on the standardization of validation documents, with and without trace links and the script.
Question 4.3:	Is it easier to create validation documents?
Metric 4.3.1:	(B) Opinions and perceptions on the ease of creating validation documents, with and without trace links and the script.
Metric 4.3.2:	(B) Number of standardized parts that are now automatically generated. *(Quantitative metric)*
Note:	*Cost metrics for creating the trace links needed for this traceability goal is covered by the metrics for question 1, 2 and 3.*
Scenario 4:	Given a cgMLST schema that is ready to be released, it should be easier to automatically generate standardized parts of the validation document. The validation document should of higher quality and more standardized.
Goal 5:	Improve change control of validation documents with the help of trace links from a bioinformatician's point of view.
Rationale:	Part of change control is doing a change impact analysis, where for example an analysis of how artifact A is affected by changes in artifact B takes place to decide if/how artifact A should be updated to take the new changes into account. Trace links can thus be used to improve change control of a validation document by making it clear which exact artifacts were used to create the validation document. If one of the artifacts is changed, a change impact analysis on the validation document can be done and the document can be updated accordingly. In addition, existing trace links to other artifacts could make doing necessary updates to the validation document easier. The artifacts that could affect the validation documented are listed in the scenarios below.
Scenario 5.1:	If the script for generating validation documents is changed, the trace links should help deciding which validation documents should be updated by making it clear which version of the script each validation document was created with.
Scenario 5.2:	If the the script for generating validation documents is changed and it has been decided that a validation document needs to be updated, the trace links should help updating the validation documents by making it easier to access the artifacts needed to generate a new version (e.g. the benchmark results).

Table 5.3: Traceability Goals, questions and metrics. *(Continuation)*

Scenario 5.3:	If a new dataset for an already released cgMLST schema is properly implemented and benchmarked, the trace links should help updating the validation documents by making it easier to access the artifacts needed to generate a new version (e.g. benchmark results from another dataset). (In this scenario, doing a change impact analysis first is not necessary)
Question 5.1:	If the script for generating validation documents is changed, is it easy to decide which validation documents should be updated?
Metric 5.1:	(B) Opinions and perceptions on the ease of deciding which validation documents should be updated if the script for generating validation documents has been changed.
Question 5.2:	If a validation document needs to be updated due to the script for generating validation documents having changed, is it easy to do so?
Metric 5.2:	(B) Opinions and perceptions on the ease of updating a validation document with a new script version.
Question 5.3:	If a new dataset for an already released cgMLST schema is properly implemented and benchmarked, is it easier to update the validation document accordingly?
Metric 5.3:	(B) Opinions and perceptions on the ease of updating a validation document with a new dataset and associated benchmark results, with and without trace links and the script.
Note:	*Cost metrics for creating the trace links needed for this traceability goal is covered by the metrics for questions 1 and 3.*

5.2 Steps 3 & 5 - Derive TIM and Assess Traceability Goals against TIM

According to TracIMo, a TIM is to be created in step 3 by using the results of steps 1 and 2. Later in step 5, the traceability goals are to be assessed against the TIM to make sure that the TIM supports the traceability goals. Since this step can lead to changes in the TIM, it is convenient to combine it with step 3. The TIM is in other words created in iterations, with changes being made as flaws and missing links are discovered in step 5.

Based on the traceability goals, the initial TIM in Figure 5.5 was derived in collaboration with the interviewed bioinformaticians and the CTO. Each arrow in the TIM represents a trace link that the traceability strategy should include. Note that the arrows are directional, marking the primary trace link direction in which the user should be able to trace the links in order to fulfill the traceability goals. For example, the arrow between the benchmark results and the dataset means that the associated dataset should be traceable from the benchmark results, but not necessarily vice-versa. Although being able to follow the trace link in the opposite direction

could be useful, it might be challenging to add this depending on how the trace links are later implemented. If an arrow points in both direction, it means that the trace link should be bidirectional, i.e. that the user should be able to trace in both directions. Although the initial TIM did not include any bidirectional trace link, updated versions did. The TIM also includes labels on the trace link, which describe link semantics, and added information about each artifact type that distinguishes from artifacts of the same type. The numeric relationship, i.e. cardinality, between artifact types are also shown on the trace links using conventional unified modeling language (UML) notations. For example, the notations on the trace link of the TIM between cgMLST schema and benchmark results mean that each benchmark result is connected to one and only one cgMLST schema, while each cgMLST schema can be connected to zero or more benchmark results.
The traceability goals (TG) were then systematically assessed against the TIM by checking that each traceability goal was supported by the trace links and that no trace link is superfluous, as seen in Table 5.4. This assessment lead to modifications being made to the TIM, as seen in the updated TIM in Figure 5.6. Rationales behind each modification are explained in the tables.

Table 5.4: Assessment of Traceability Goals against the TIM. Traceability goals have been shortened to improve readability.

Traceability Goal	**How the TIM supports the goal**
TG 1: Improve accessibility from a validation document to the exact versions of artifacts that went into the generation and validation of the cgMLST schema.	Trace links to cgMLST schema, benchmark results and script for generating validation documents allow for easy access to these artifacts. However, the trace link to benchmark results was later removed due to the assessment of Goal 4 (see below). Trace links to the other artifacts were not added since there already are trace paths going to them via the cgMLST schema and benchmark results. For example, to access the exact associated pipeline for generating cgMLST schemas from a validation document, one can do so via the cgMLST schema.
TG 2: Improve accessibility from a cgMLST schema to the exact versions of artifacts that went into the generation of the schema.	Trace links from the cgMLST schema to the pipeline for generating cgMLST schemas and reference genomes allow for easy access to these artifacts. During this assessment, it was discovered that a trace link from the schema to itself was missing and therefore added in the final TIM.
TG 3: Improve accessibility from a benchmark results to the exact versions of artifacts used to produce the benchmark results.	Trace links to the cgMLST schema, dataset, cgMLST analysis pipeline and benchmarking scripts allow for easy access to these artifacts.

Table 5.4: Assessment of Traceability Goals against the TIM. Traceability goals have been shortened to improve readability. *(Continuation)*

TG 4: Improve creation process of validation documents by allowing standardized parts to be automatically generated with the help of a script and trace links to related artifacts.	Being able to trace from the cgMLST schema to the benchmark results is important for this goal since locating the benchmark results needed to create the validation document is more difficult without it. However, as seen in Figure 5.5, the trace link between these artifacts only specified traceability from the benchmark results to the cgMLST schema as important. Due to this assessment, this trace link was thus updated to be shown as bidirectional. Consequently, the trace link from the validation document to the benchmark results was removed to avoid having superfluous trace links (the benchmark results can now be traced from the validation document via the cgMLST schema). The trace link from the validation document to the cgMLST schema is also important for this goal.
TG 5: Improve change control of validation documents with the help of trace links.	All trace paths stemming from validation documents will support change impact analyses since it will facilitate pointing out affected artifacts.

Figure 5.5: Initial Traceability Information Model.

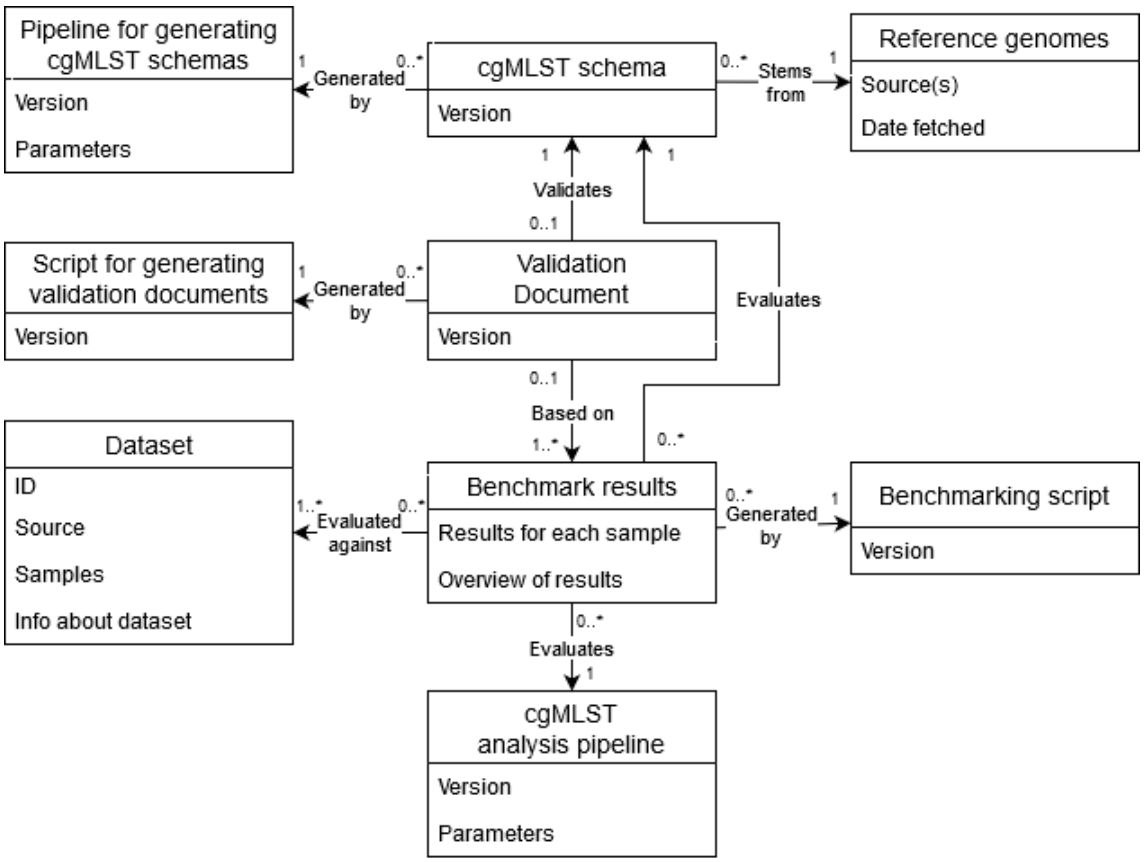
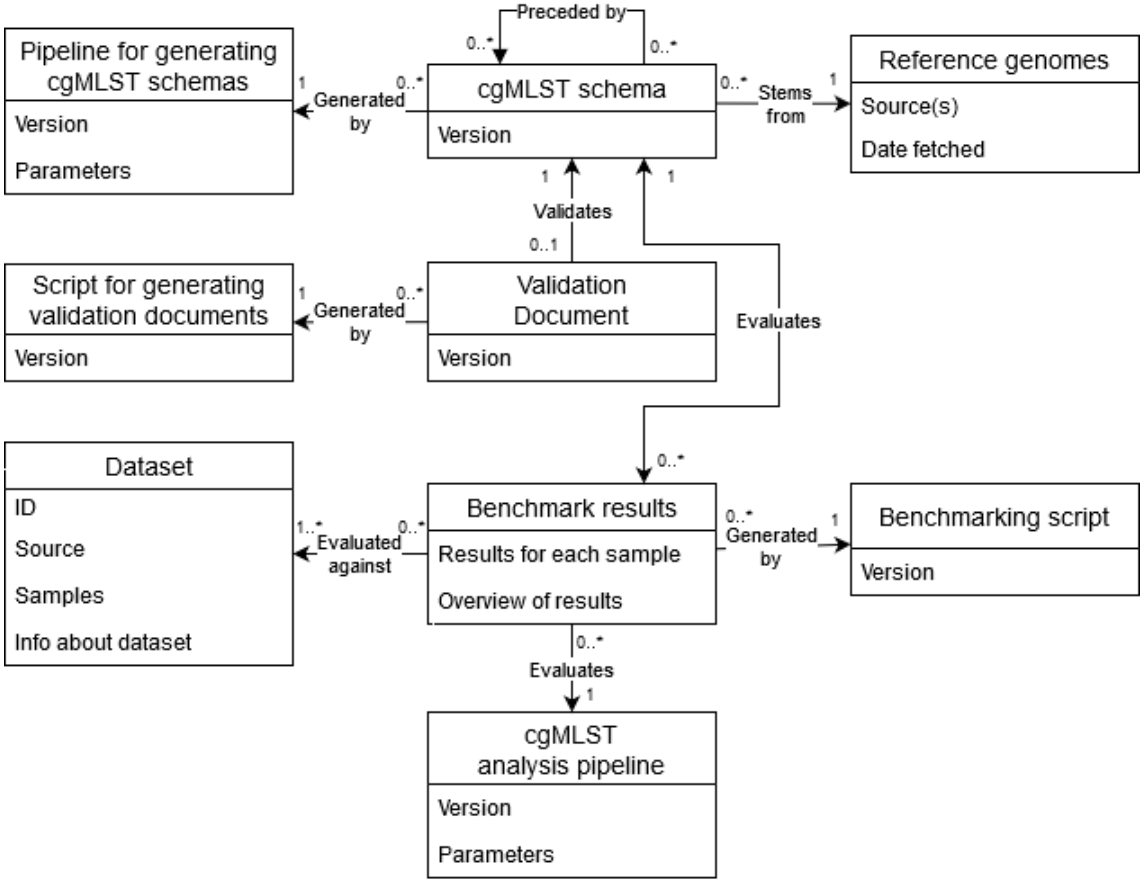


Figure 5.6: Updated Traceability Information Model.



5.3 Step 4 - Assess the Process Goals against the Traceability Goals

To assess the process goals (PGs) against the traceability goals, they were mapped in Table 5.5 to ensure that at least one traceability goal covers each process goal that requires traceability. In addition, it was checked that each traceability goal supports at least one process goal to ensure that no traceability goal was superfluous, i.e. not adding any value to the development process.
Thereafter, these process goals were assessed against the general process goals (GPG) to make sure that there were no contradictions. All GPGs were not required to be supported by the PGs though, since this was never the intended scope of the traceability strategy. GPGs that might negatively be affected by the planned traceability are discussed in Table 5.6. GPGs not included in the table are seen as either unaffected or only positively affected by the PGs. This assessment is not originally included in TracIMo since there wasn't a separation between general process goals and process goals for a specific development area. Nonetheless, this assessment was seen as important to avoid contradictions.

Table 5.5: Assessment of the Process Goals against the Traceability Goals. The table shows which traceability goals support each process goal. Goals have been shortened to improve readability.

Process Goal	Supported by Traceability Goals
PG 1: Improve process for creating and validating cgMLST schemas by having all artifacts necessary to reproduce and verify the contents of a validation document uploaded.	**TG 1:** Improve accessibility from a validation document to the exact versions of artifacts that went into the generation and validation of the cgMLST schema. **TG 2:** Improve accessibility from a cgMLST schema to the exact versions of artifacts that went into the generation of the schema. **TG 3:** Improve accessibility from a benchmark results to the exact versions of artifacts used to produce the benchmark results.
PG 2: Improve standardization and quality of validation documents and facilitate the creation of them.	**TG 4:** Improve creation process of validation documents by allowing standardized parts to be automatically generated with the help of a script and trace links to related artifacts.
PG 3: Improve change control of validation documents.	**TG 5:** Improve change control of validation documents with the help of trace links.
PG 4: Maintain lightness of main repository by keeping out bulky artifacts that are not needed by any product.	
PG 5: Reduce number of benchmarking scripts used to facilitate maintenance and continued improvement of benchmarking capabilities.	
PG 6: Increase number of datasets that are properly implemented so that benchmarking takes less time.	

Table 5.6: Assessment of the General Process Goals against the Process Goals.

General Process Goals	**Process Goals**	**Assessment**
GPG 4: Increase the speed of the test suite from a developer's point of view.	**PG 5:** Reduce number of benchmarking scripts used to facilitate maintenance and continued improvement of benchmarking capabilities from a bioinformatician's point of view. **PG 6:** Increase number of datasets that are properly implemented so that benchmarking takes less time from a bioinformatician's point of view.	Although PG 5 may increase the amount of time it takes to run a benchmarking script, this could in the long-run be more than compensated by not having to develop new benchmarking script for each dataset. In addition, PG 6 also supports GPG 4.
GPG 5: Maximize cost-benefit of created artifacts by avoiding to create requirements and documentation (or other artifacts) that don't have a positive cost-benefit from the from the development team or customers' point of view.	**PG 1:** Improve process for creating and validating cgMLST schemas from a bioinformatician's point of view by having all artifacts necessary to reproduce and verify the contents of a validation document uploaded so they don't get lost.	The process of uploading all artifacts could be costly, hence it is important that this is done as seamlessly as possible and that the bioinformaticians in fact do benefit from them being uploaded.

5.4 Step 6 - Derive Traceability Process

The purpose of this step is to define how trace links are to be created, maintained and ultimately used in the development process. The input for this step were all goals, the TIM, the existing development process, the scenarios, numerous discussions with the interviewed bioinformatician and some input from other developers. In addition, the CTO had a number of general preferences regarding how trace links should be created that were also taken into account. These preferences, listed below, had been identified from the interviews with the CTO and QA/RA director and later member checked.

1. When possible, trace links should be achieved by uploading artifacts in *modular structures*, i.e. where associated artifacts are saved close to each other in an organized folder structure so that the bioinformatician can implicitly understand the association between artifacts. Usage of external tools for creating trace links should be avoided.
    - *Rationale:* A modular structure is preferable compared to using an external tools since developers prefer using the tools they already use for development, e.g. directories, rather than having to jump back and forth between different tools. If traceability were in another tool, there would be the risk that developers forget about it and never use it. While reaching for an artifact, modular structures also make it easy to passively notice which related artifacts exist.
2. Minimize extra steps for creating trace links by using automation when possible.
    - *Rationale:* If creating trace links is too much of a hassle, developers will not create them. Creating trace links should thus be as seamless and have as little friction as possible. This applies even if a modular structure is to be used to establish trace links by automatically uploading and organizing the artifacts since choosing the right location to save an artifact also takes time.
3. Decrease decay of traceability with additional tests.
    - *Rationale:* This can offer an extra layer of insurance that all artifacts have the trace links they should. For example, one could have tests that check if a collection of requirements all have trace links to verification, meaning that developers don't have to check this themselves.

Due to the second preference, it was decided that trace links should automatically be created as artifacts are created, instead of having them be created afterwards. The process for how trace links are to be created in parallel with the creation and validation of cgMLST schemas was then derived in collaboration with the bioinformatician and is modelled in Figure 5.7. The central idea is for the task generating an artifact to also create the trace link from the generated artifact to the other existing artifacts as stated by the TIM. If the created artifact should be connected to another artifact that does not exist yet, then this trace link should be created when the other artifact is generated.

For example, the task named "Generate cgMLST schema & add trace links" is meant to not only generate the cgMLST schema but also create the trace links from the

schema to the pipeline for generating cgMLST schemas, the reference genomes and preceding cgMLST schemas (if such exist). It is in other words not meant to create the trace links to the benchmarking results (since they have not yet be generated). The new process is explained more in detail below.

5.4.1 CgMLST schemas

The distinction between schemas that have been released and unreleased schemas was not shown in any of the previous TIMs since they would have the same trace links (except for that validation documents are only connected to released cgMLST schemas). However, after having considered the process for creating the trace links, it became clear that this distinction is important. This is because only released schemas should be uploaded to the main repository (since bulky artifacts not needed by customers should not be uploaded), but all created schemas, regardless if they are later released or not, will have trace links to the other artifacts. To avoid having to move a schema ready to be released and having to adjust existing trace links accordingly, it was determined that leaving all schemas where they are (outside the main repository) and creating a bidirectional trace link to a copy of the schema in the main repository (which is the one used by the customers) was the best solution. The TIM was thus updated to take this distinction into consideration, as seen in Figure 5.8. Schemas that have been released and added to the main repository will as of now be referred to as released cgMLST schemas, while the original schemas containing the trace links to the reference genomes and pipeline for generating cgMLST schemas will just be referred to as cgMLST schemas.
Following the central idea, this trace link is to be created in the task called "Add cgMLST schema to main repo & created trace link", where the cgMLST schema is created.

5.4.2 Benchmarking script

A significant change in the development process is that instead of a bioinformatician creating and tweaking benchmarking scripts for each schema that is to be benchmarked, a single benchmarking script is to be used. A bioinformatician shall be able to specify what and how to benchmark when running the script without any hard coding, which explains why the step "Create / Tweak Benchmarking script" is no longer needed.
Following the central idea, in the task "Generate benchmark results & create trace links", trace links from the generated benchmark results to the benchmarking script itself (since it may happen that the benchmarking script is updated), the dataset and the cgMLST analysis pipeline are to be created. Furthermore, the script shall automatically upload the benchmark results somewhere where they won't get lost instead of just storing them locally.

5.4.3 Creation of validation documents

Another significant change is how the validation document is created. Before, the input for this task was the released cgMLST schema and the notes taken by the

bioinformatician. In the new process, the task's input is the released cgMLST schema, the script for generating validation documents and the benchmark results. The reason why the notes have been omitted is because with the new script for generating validation documents and trace links, looking at notes shouldn't be as needed. With proper trace links to the benchmark results, the bioinformatician should easily be able feed them to the script to produce the validation document.
The new process does not discourage note taking and the bioinformaticians are free to take notes as they wish, which was seen as important for the bioinformaticians and the CTO. As stated in the goals, the aim is simply for all relevant artifacts to be accessible, which should naturally decrease the need for bioinformaticians to be diligent taking notes. As earlier mentioned, it is also not possible for the script to generate the entire validation document since there are steps in the validation process that are done manually, meaning that note taking probably won't disappear completely. Nevertheless, notes have been omitted from the process model due to the fore-mentioned reasons and to reduce clutter.

5.4.4 Retroactively creating trace links for already released cgMLST schemas

Since trace links are only created while creating of artifacts and not retroactively, this process is not suitable for creating trace links to artifacts related to an already released cgMLST schema. Nonetheless, it is important for the traceability goals that artifacts related to already released cgMLST schemas also have trace links. Thus, a separate process for creating trace links for such artifacts must also be derived. To avoid manually having to create trace links and make use of the new process, certain artifacts should be recreated. With the assumption that the new process makes good use of automation, this was seen as the most effective method. Note that this does not mean that the released cgMLST schemas should be replaced with new ones. This would be very inefficient and go against the rule of not replacing a cgMLST schema unless absolutely necessary. The process model in Figure 5.9 depicts how a trace links for artifacts related to a released cgMLST schema should be created.
To further explain this process, an explanation of some of the steps and the rationales is here given.
The process starts with the bioinformatician searching for the exact reference genomes and parameters on the pipeline for generating cgMLST schemas that were used to generate the released cgMLST schema with the help of old the validation document and notes (if available). Since no trace links to these artifacts will exist, this task will have to be done manually.
Next, using the reference genomes, parameters and the current version pipeline for generating cgMLST schemas, a cgMLST schema identical to the previously released shall be generated while simultaneously creating trace links to the mentioned artifacts. It will not be possible to create trace links to the exact version of the pipeline for generating cgMLST schemas that was originally used though (according to the bioinformaticians, the version of the pipeline should not have an influence on the generated cgMLST schema though). Furthermore, trace links to preceding cgMLST

Figure 5.7: Process model for generation and validation of cgMLST schemas while creating trace links. Red dotted lines represent trace links.

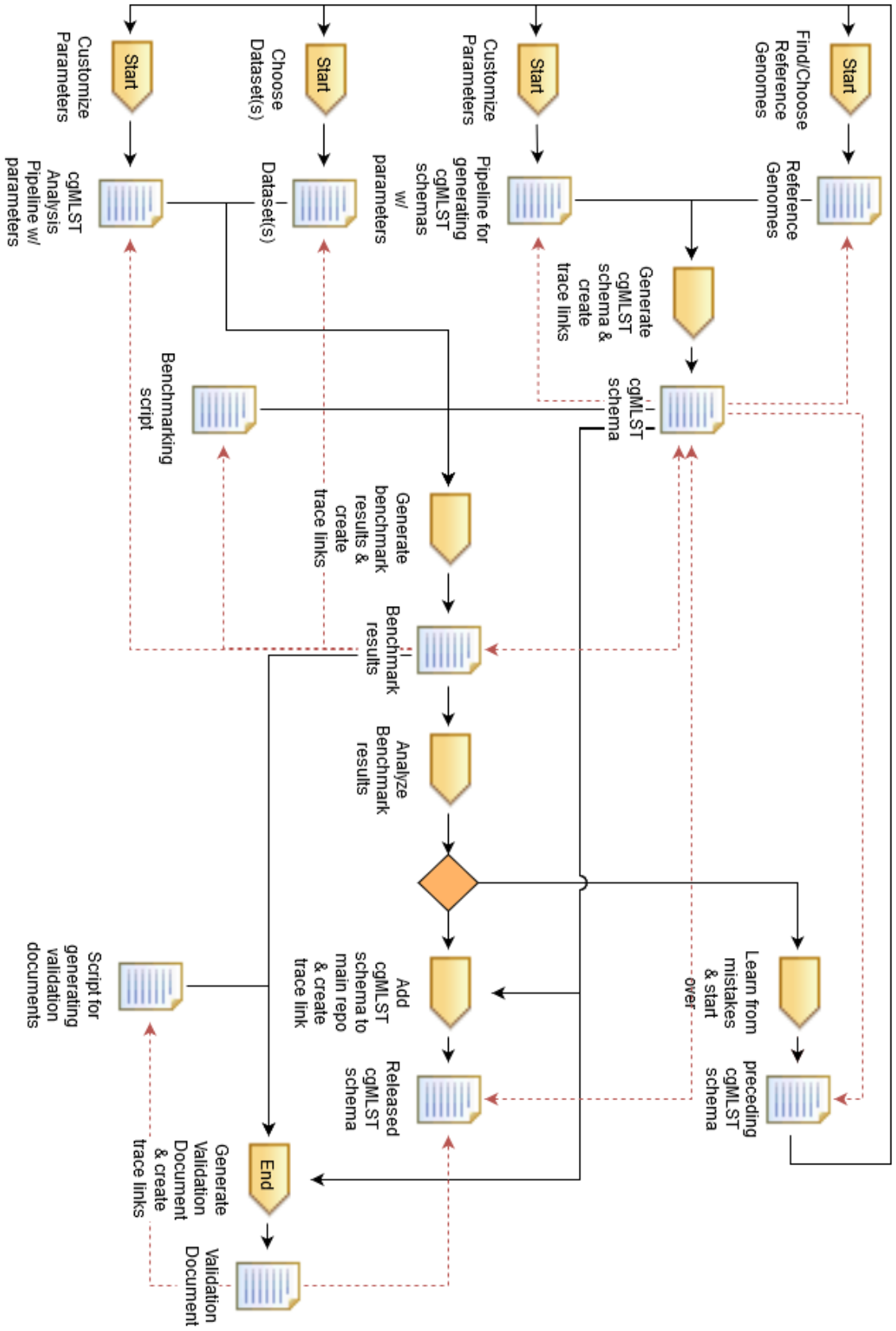


Figure 5.8: Final Traceability Information Model with the distinction between a cgMLST schema and a released cgMLST schema.

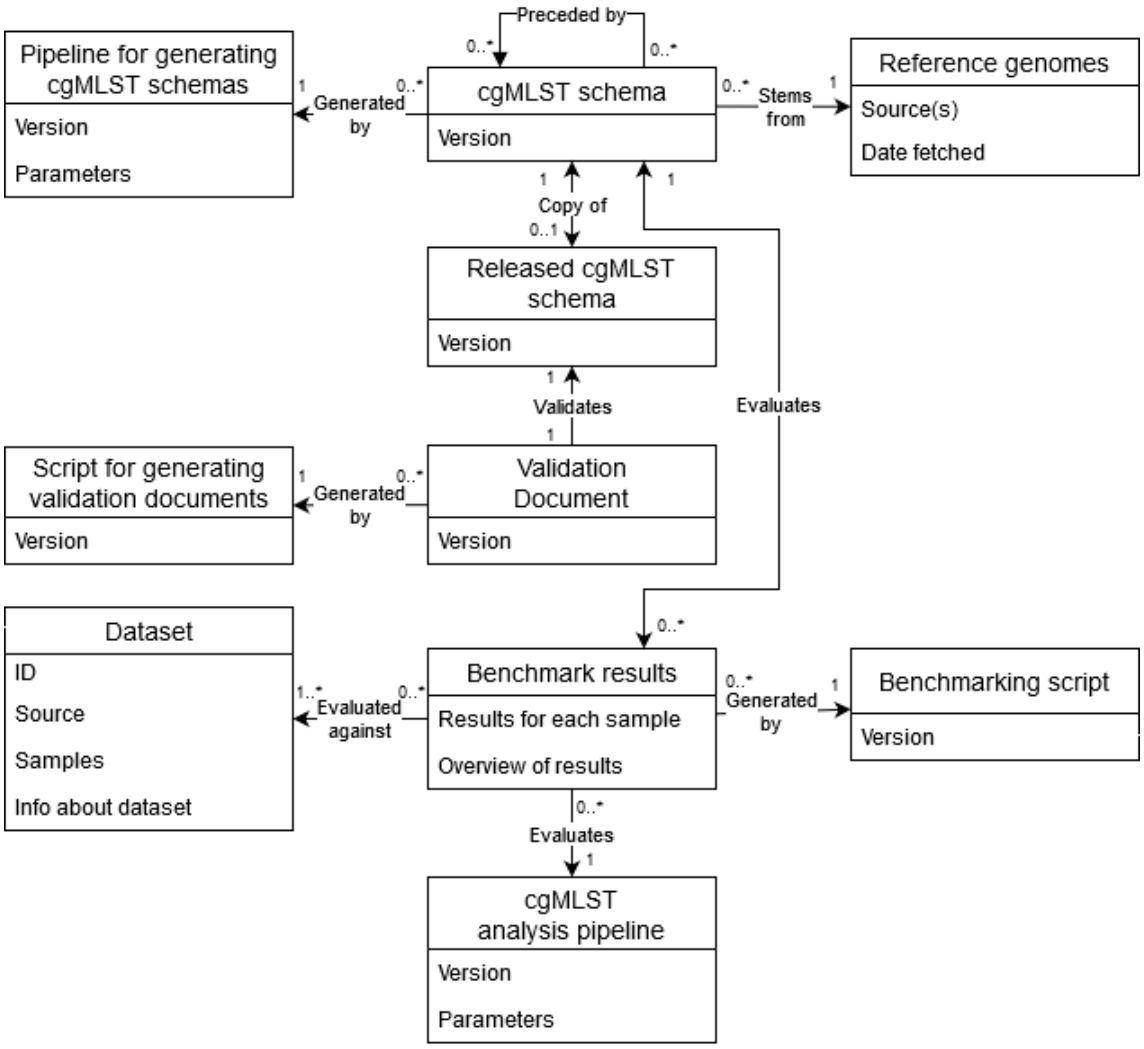
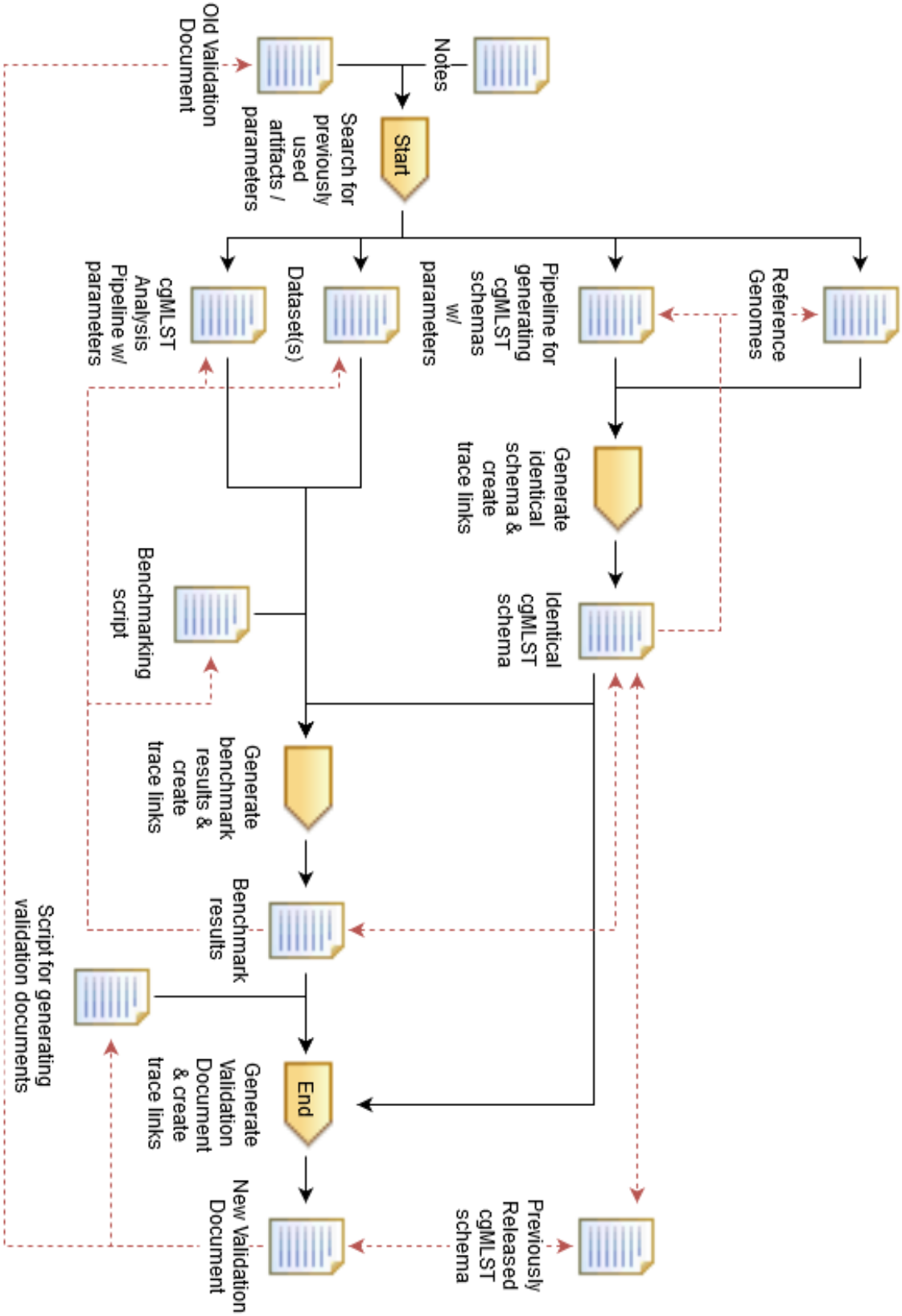


Figure 5.9: Process model for retroactively creating trace links to artifacts related to an already released cgMLST schema.



schema versions won't be created since these artifacts have generally not been kept and finding/recreating them would take too much effort, if even possible.
Thereafter, using the available dataset(s), cgMLST analysis pipeline, and the new benchmarking script, new benchmark results shall be generated while simultaneously creating the trace links. This task is in other words to be done in the same way as in the normal process depicted in Figure 5.7. In this case, it is not important for the original artifacts be used and identical benchmark results to be produced. What's important is just for the results to show that the schema performs well.
Lastly, a new validation document is to be created using the script for generating validation documents, while at the same time creating the necessary trace links. This is also to be done in the same way as in normal process. To avoid any information from the old validation document getting lost, a trace link to the old validation document should also be created.

5.5 Step 7 - Select and customize tool

In this step, a description of how we selected and customized the tools used to implement the derived traceability process is given. Due to the first preference listed in step 6, it was decided that no external tool should be used. The central idea was for the kind of tools that the bioinformaticians already use to generate and validate schemas to automatically create the trace links. In other words, pipelines and scripts, both new and existing, were to be enhanced as to also create the trace links. This also meant that the trace link themselves should be stored using the existing tools, e.g. using modular structures as suggested in the first preference, within S3 and the main repository.
Since the designed tools were not fully developed by the end of this thesis, this step is divided in two sections, one where the design of the tools are described and another where the state of the tools at the end of this thesis are described.

5.5.1 Design of tools

After various discussions with the interviewed bioinformatician and several iterations, it was decided that the artifacts and trace links should be stored and implemented using the tools as depicted in Figure 5.10. The enhanced TIM shown in Figure 5.11 gives an overview of where each artifact type shall be stored. The directionality of the trace links represent the only directions in which the user will be able to trace them given the selected tools. Note that certain trace links are now bidirectional whereas in earlier TIMs they weren't. This is simply due to the fact that the chosen implementation provides this bonus, it is not related to the traceability goals.
Below, a more in depth explanation of the process using the selected tools is presented.

- **cgMLST schema, Pipeline for generating cgMLST schemas and Reference genomes:** S3 has been chosen as the most suitable location to store all cgMLST schemas (instead of having them spread out between the S3 and

Figure 5.10: Process model using scripts and pipelines for creating trace links.

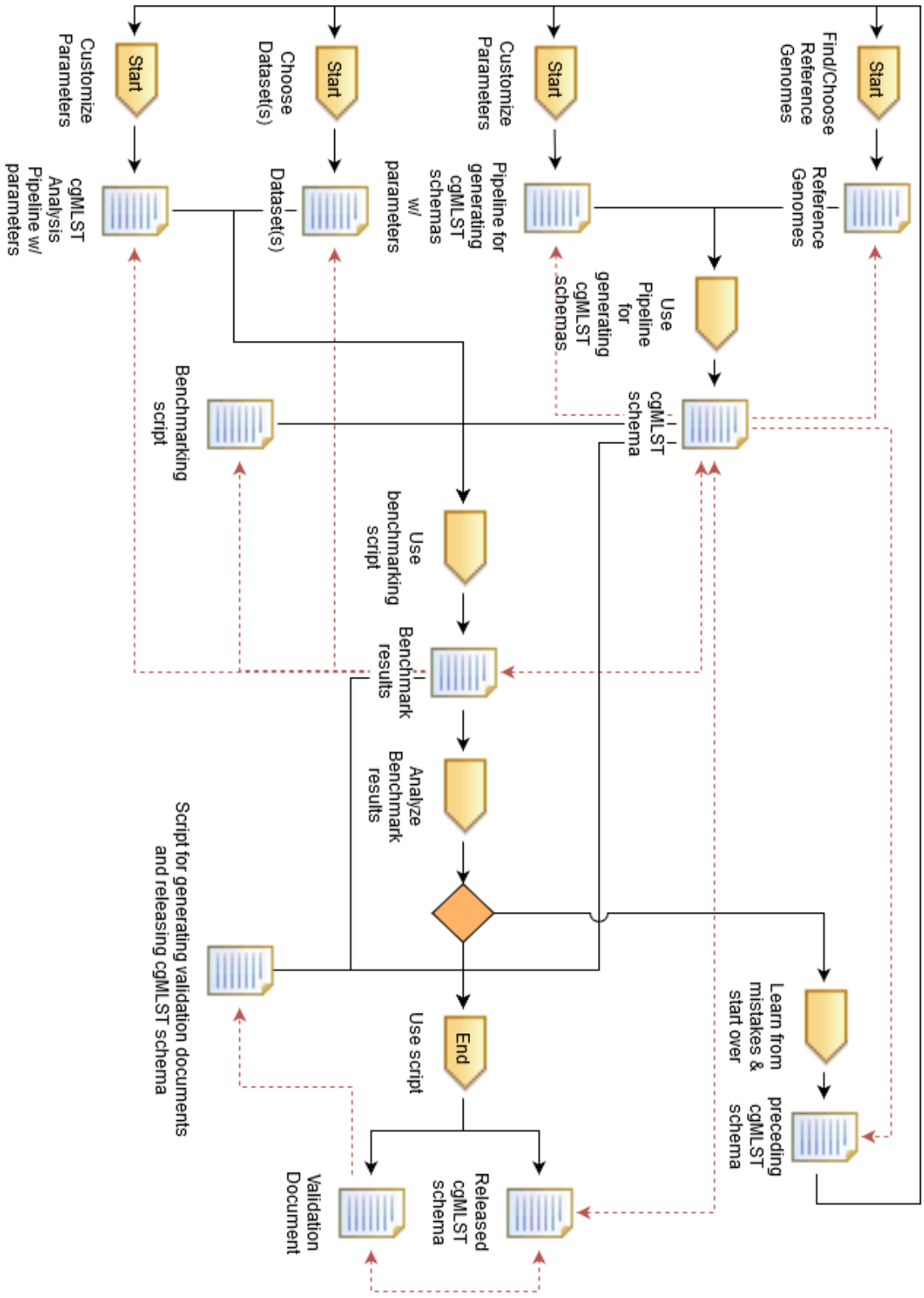
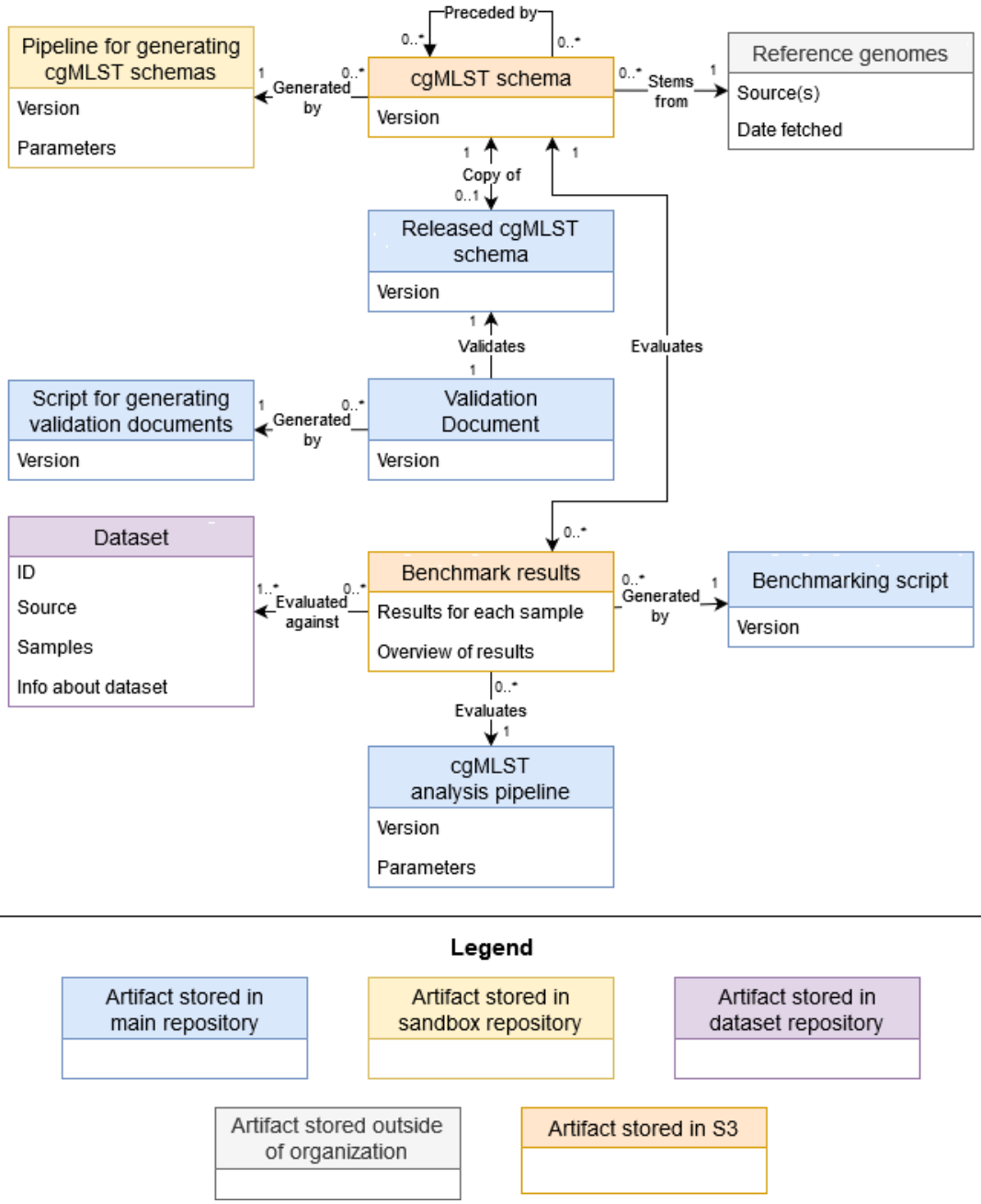


Figure 5.11: TIM where artifact types have been color-coded to illustrate where they will be stored.



being locally saved). The pipeline and the reference genomes remain where they were previously stored.

Following the central idea, the pipeline for generating cgMLST schemas was to be enhanced so that it not only generates the schemas, but also automatically uploads them to S3 and creates the trace links to the pipeline itself, preceding cgMLST schemas as follows.

Trace link to preceding cgMLST schemas, are established by having a modular structure in S3, where all schemas for a certain pathogen are saved in the same directory as shown in Figure 5.12. A bioinformatician can in other words simply browse the directory to find preceding as well as succeeding schema by comparing their version numbers. Note that this trace link is now bidirectional, which naturally happens whenever a modular structure is used to establish trace links.

As for the remaining trace links, the bioinformaticians proposed modifying the pipeline so that it would generate additional accompanying files containing information regarding which pipeline version, parameters and reference genomes were used. These accompanying files would be stored in a folder together with the schema to make it clear which schema they belong to. To show which version of the pipeline for generating cgMLST schemas was used, the commit hash shall be documented. As for the reference genomes, their unique IDs

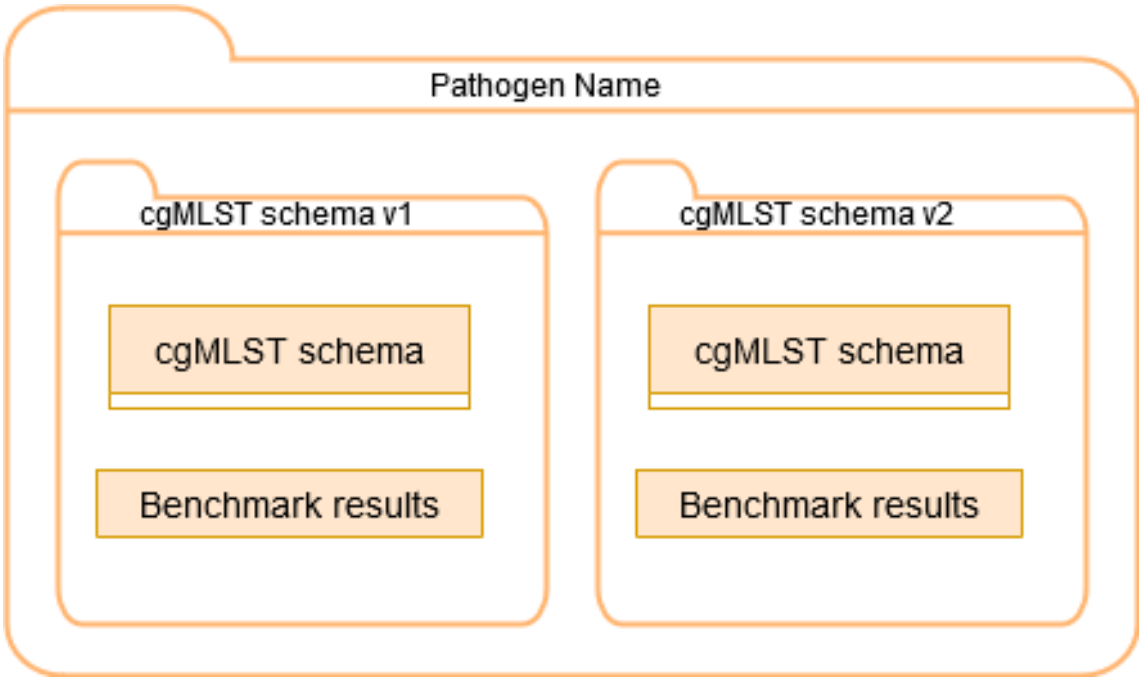
- **Released cgMLST schema:** The script for generating validation documents is also to be used for releasing cgMLST schemas, i.e. uploading a copy of the schema in S3 to the main repository. As seen in Figure 5.10, generating a validation and releasing a cgMLST schema are to done simultaneously, instead of two separate tasks as shown in Figure 5.9. This decision was made for convenience and to make this task easier.

  Released cgMLST schemas must be in the main repository for customers to use it. Thus, since the cgMLST schema and the released cgMLST schema are stored in different tools, using a modular structure to create trace links is not possible. However, since these artifacts are actually identical, it made sense for the script to assign the same names to the cgMLST schema and the released cgMLST schema. A bioinformatician can thereby easily see which schemas are associated by looking at their names.
- **Validation Document, Script for generating validation documents:** Instead of saving the validation documents in Google Drive, they are now to be saved in the main repository close to the associated released cgMLST schemas. This is in line with the seventh general process goal, stating that documentation should be located close to the related code implementation, and makes it easier to create trace links between the documents and the released cgMLST schemas. Now that they are stored within the same tool, a modular structure can be used to connect the artifacts, which also explains why the trace link is now shown as bidirectional in the TIM.

  The trace link from the validation document to the version of the script for generating validation documents is implicitly created by git whenever the validation document is pushed to the main repository.

  In order to create a new validation documents for an already release cgMLST

Figure 5.12: Conceptual representation of the modular structure used in S3 to create trace links between cgMLST schemas and associated benchmark results.



schemas, the script needs to be able to generate a validation document without also releasing a cgMLST schema. When this is done, a copy of the old validation document (in Google Drive) should manually be uploaded to the main repository next to the new document. Developing a script for this task was considered but deemed to not be cost-efficient since the task is very simple and won't be done very often.

- **Benchmark results, Dataset, Benchmarking script and cgMLST analysis pipeline:** The benchmark results have been consolidated to S3. This is to avoid benchmark results getting lost, which previously was an issue.
  The benchmarking script shall automatically upload the results to S3 once they have been created following a modular structure to create a bidirectional trace link to the associated cgMLST schema as shown in Figure 5.12. The benchmarking script shall also include information within the benchmark results about the dataset(s) and the exact versions of the cgMLST analysis pipeline and benchmarking script, thereby creating trace links from the results to these files. However, for the benchmarking script to be able to run a dataset, the dataset must be properly implemented. This was necessary for the benchmarking script to be able to create the trace links.

Some could argue that the trace links created by the developed tools don't really count as trace links since explicitly links aren't created, but rather the use of modular structures and documentations that allow users to access or retrieve sufficient information about associated artifacts. According to the the book *Software And Systems Traceability* [4] though, the definition of a trace link is *"a specified association between a pair of artifacts"*. Assuming that the association may take any form as long as it somehow connects the artifacts, we therefore still choose to define these

connections as trace links.

5.5.2 State of tools

The interviewed bioinformatician got to work developing the enhanced scripts, pipelines and setting up the modular structures within S3. Only prototypes of the tools were developed, although the bioinformatician will continue working on them after the conclusion of this thesis. Including the time spent discussing how to derive the traceability process and selecting which tools to use, the interviewed bioinformatician had spent approximately one month of full-time work on this project. Below, a summary of the state of each script and pipeline is given.

- **Pipeline for generating cgMLST schemas:** An important thing that still needed to be improved was the flexibility of the pipeline, since the functionality to exclude reference genomes from NCBI RefSeq (e.g. mislabeled ones) was not yet implemented and would require a lot of time to complete. This meant that generating a new, well-performing cgMLST schema that could actually be released was not be possible. In addition, generating an identical copy of an already released cgMLST was also not be possible since new reference genomes that have been added to NCBI RefSeq after the schema was originally created can't be excluded. Other than this, the tool functioned as planned.
- **Script for generating validation documents and releasing cgMLST schemas:** The prototype of the script for generating validation documents and releasing cgMLST schemas was limited in a number of ways. Firstly, it did not actually upload the generated document to the main repository, meaning that the trace link to a released cgMLST schema could not be shown. Fixing this would be easy and the bioinformaticians did not need to see the trace link to feel comfortable evaluating it. Secondly, it generated the validation documents as .txt files, which doesn't offer the formatting functionalities needed to create a validation document that is presentable for customers. Thirdly, the option to generate a validation document without uploading the cgMLST schema to the main repository was still missing. This option needs to be added if new validation documents are to be created for already released schemas. Finally, the standardized part containing an overview of how a cgMLST schema did not include the parameters used in the pipeline for generating cgMLST schemas. A screenshot showing how a validation document generated by the script is shown in Figure 5.13.
- **Benchmarking script:** The benchmarking script had all the planned functionality. Only further performance optimization (e.g. making it run on a remote server instead of a bioinformatician's laptop) was needed for it to be used in reality.

Figure 5.13: Validation document containing both standardized parts that has been automatically generated using the script for generating validation documents. The first five sections contain the overview of how the cgMLST schema was created. The first two sections here explain which seed genome and reference genomes were chosen and how. The next three sections explain how the pipeline for generating cgMLST schemas chose the core genomes., The remaining sections contain the overview of the benchmark results. Manually created parts have not been added.

```
All complete genomes available from NCBI were retrieved on 2020-08-28 (n=30)
and used as reference genomes.

The seed genome had RefSeq assembly accession GCF_001571545.2. Genes
from the seed genome with identity higher than 90.0%
homology threshold between each other were removed from being target genes.

For a target gene to be considered as present in a reference
genome, it was required to match above 90.0% identity cutoff.

For a gene to be accepted as a core gene into the generated schema,
it was required to be present in more than 95.0% of reference genomes.

The resulting cgMLST schema yielded 2055 core genes representing 39.6% (2055 / 5191)
of the coding genes from the seed reference genome.

Benchmark results:
Dataset Mean    Median  Number of STs   Number of samples       App version
mikker  98.99   99.2    1       69      2020-07.5-17-g3e60bffc-dirty

Number of samples with fraction of core below 95%: 2 out of 69 samples
Number of samples with low sequencing depth: 2 out of 69 samples
escherichia_coli_validation_document.txt (END)
```

5.6 Step 8 - Deploy process and tool

Once the prototypes were finished, a meeting with the interviewed bioinformatician and another bioinformatician was held for the interviewed bioinformatician to demonstrate the new process. The reason why only one other bioinformatician was invited was because the two other bioinformaticians don't generate any cgMLST schemas (they develop other things), hence they would never need to use the process and tools. During the meeting, the other bioinformatician got the chance to experience how it was to generate and benchmark a dummy cgMLST schema and then generate a validation document using the new process and tools (without actually pushing anything to the main repository) [1]. He could then also see how the trace links worked.
Due to the discussed limitations of the pipeline for generating cgMLST schemas, it was not possible to use the process and tools to create and release a cgMLST schema for a new pathogen or retroactively create trace links for an already released cgMLST schema before the evaluation. Although using the process and tools for a cgMLST schema that has already or would be released would have been preferable, using them for a dummy schema was sufficient for the bioinformaticians to get an impression on the process and how the trace links could be used. Holding off with the evaluation until the tools were fully deployed was considered but ultimately the prototypes were seen as functional enough.

5.7 Step 9 - Evaluate process and tool

To evaluate the process and tools, two different meetings were held. First, the two bioinformaticians who would end up using them were interviewed in a group interview. The purpose of the group interview was to go through the entire process and tools in detail to evaluate whether the traceability goals had been achieved and identify their weaknesses. Thereafter, the bioinformaticians, the CTO and the QA/RA director were invited to a focus group. The purpose of the focus group was to present the deployed process and tools to the CTO and QA/RA, offer them the opportunity to ask questions and give feedback, and bring about discussion on how traceability could benefit the company going forward.
The reason why two separate meetings were organized was because it was not suitable to include the CTO and QA/RA director in the group interview where the process and tools would be evaluated in detail (considering that they would not be using them). To get feedback from them, a focus group was seen as more suitable.

5.7.1 Group Interview

Based on the traceability goals, questions, metrics and measurement plan from step 2 and the deployed traceability solution, questions for the group interview seen in

[1] The bioinformatician first tried using the tools himself but they failed due to minor bugs. Nonetheless, he could initiate them and see the intended output by accessing the artifacts that were uploaded by the interviewed bioinformatician.

Appendix A.1 had been prepared. The interview questions were mapped against the goals and metrics to make sure that each one had been covered. Individual answers for each metric are shown in Appendix A.2. In this section, a summary of the gathered information is presented.

The bioinformaticians could already see progress towards improving the accessibility of artifacts. There still were functionalities missing, but they thought the tools would fulfill each traceability goal related to improving accessibility. In particular, they were very positive towards benchmark results and artifacts needed to reproduce cgMLST schemas automatically being uploaded and more accessible.

The one kind of trace link that had issues were the ones that were implemented by documenting commit hashes. Although they worked in theory, they had two weaknesses. Firstly, it took a lot of clicks for a bioinformatician to trace this kind of trace links. Secondly, if the branch where the trace link wasn't merged or rebased to the master branch properly, the commit hash could be deleted from the repository, leading to the trace link rendered useless. Alternate solutions were discussed (such as renaming files whenever they are changed and referring to the file names instead), but solving this issue was not highly prioritized since adding missing functionality was more important.

One drawback that was raised about the benchmarking script was that datasets are now required to be properly implemented. This means that effort needs to be made before a new dataset can be used for benchmarking. However, increasing the number of properly implemented datasets was an identified process goal, so enforcing this was also seen as something positive.

Since the trace links were created automatically, they viewed the effort in creating trace links as very low or none at all. The only case where noticeable effort will need to be invested in maintaining trace links is whenever a script fails to complete its task properly. This may for example be the case if the benchmarking script crashes (which may happen since benchmarking often takes several hours and requires a lot computing) and incomplete benchmarking results need to be deleted. While deleting incomplete artifacts, accidentally deleting the wrong artifacts would also lead to more efforts in maintaining trace links. A potential solution that was discussed is adding the functionality to automatically delete incomplete artifacts. As the tools are used more and more, the bioinformaticians also expect them to get optimized, which would reduce the chance of them failing.

The process of retroactively creating trace links, creating new validation documents for already released cgMLST was for the most part seen as feasible although in some cases impossible due to required artifacts being lost and too difficult to recreate. However, the bioinformaticians were unsure about the cost-benefit of doing this due to the amount of effort needed and the reduced benefit compared to creating validation documents for new schemas.

Although the script for generating validation documents and releasing cgMLST schemas was in a early state and can't be used in reality yet, some improvements in the process of creating validation documents could already be observed. Thanks to the overview of the benchmarking results being automatically generated, this process is easier and more standardized. When it comes to quantitatively measuring how many standardized parts can now be automatically generated, both standard-

ized parts can now be automatically generated, which corresponds to seven sections not having to be written manually. Since a bioinformatician no longer needs to manually calculate means and medians, there is also the reduced risk of human errors being made, which increases the quality of the validation documents. However, in terms of the formatting and presentation, the quality is seen as inferior to what a bioinformatician could do manually on Google Docs. A file format other than .txt, e.g. Markdown, will need to be used to come closer in matching Google Docs' functionality.
During the group interviews, it was discovered that the bioinformaticians are actually very unlikely to update a validation document only because the script for generating validation documents is updated. The assumption is that if a bioinformatician releases a validation document, it means the bioinformatician is satisfied with it and won't update it unless new relevant information is revealed. Thus, evaluating improvements in change control caused by changes to the script was no longer seen as meaningful (questions 5.1 and 5.2). However, the ease of updating a validation document with a new dataset and associated benchmark results (question 5.3) is still a relevant question to evaluate. A challenge that first needed to be tackled was how to use the script to update validation documents that are partially automatically and manually created. Possibly, the manually created parts would need to be recreated. The bioinformaticians thought it was too early to address this challenge though, they first wanted to use the script for a real validation document. Putting this challenge aside, updating a validation document was now seen as easier thanks to relevant artifacts being more accessible and being able to use the script to create standardized parts.

5.7.2 Focus Group

The focus group started with a short presentation conducted by the researcher for the CTO and QA/RA director to understand the deployed traceability strategy and the answers collected from the group interview. After the presentation, the CTO and QA/RA were given the opportunity to ask questions and give general feedback on the traceability process and prototype tools, which was overall positive.
Next, the participants were asked to compare the cost, i.e. effort, of implementing and using the traceability strategy to the expected long-term benefits. They were also asked if they regarded some parts of the strategy as more or less cost-beneficial. All three tools, the pipeline for generating cgMLST schemas, the benchmarking script and the script for generating validation documents, were seen as cost-beneficial but some more than others. The pipeline for generating cgMLST schema was seen as the most cost-beneficial since storing all artifacts needed to reproduce cgMLST schemas in a systematic way was a significant improvement while not taking much effort to implement. The benchmarking script took a bit more effort and was slightly less beneficial since benchmark results can be reproduced if necessary (this is to be avoided though since benchmarking takes a long time and does not look good for customers, so the script was still very beneficial). Developing the script for generating validation documents took the most effort (and will continue to take more effort) and since validation documents are not created so often also seen as less

beneficial than the pipeline for generating cgMLST schemas and the benchmarking script.
Thereafter, the participants were asked if they thought there were other development areas where implementing a similar traceability strategy would be worthwhile. The CTO stated that since all new pipelines involve benchmarking, improving other benchmarking scripts in a similar way could be a natural next step. This would also motivate bioinformaticians to properly implement datasets (since the benchmarking script only works with properly implemented datasets), which the CTO saw as something positive.
Lastly, the participants were asked if they thought the traceability strategy could be relevant in a regulated environment and what they thought was missing to make it more relevant. The QA/RA director explained that medical device regulations and other regulations are more focused on the traceability from requirements to verification of these requirements and less focused on how traceability can benefit developers. What was missing in the traceability strategy was a way to handle requirements and adding traceability from them to the verification. However, if this were to be added, the traceability strategy could be relevant since the traceability strategy improves the verification part, i.e. the benchmarking. Now that benchmark results are stored systematically, tracing requirements to the verification becomes easier. Not focusing on how to handle requirements was a conscious decision made in this project since it was not yet the time to focus on external regulations, but the topic was still discussed during the focus group. Currently, the validation documents explain how internal requirements are fulfilled but the actual requirements are not stated explicitly. One way to address the issue in the future would be to store the requirements, regardless if they are internal or external, in the validation document by stating them explicitly. By storing them in the validation document, tracing the benchmark results would already be possible.
The questions used during the focus group can be seen in Appendix A.3.

6
Discussion

In this chapter, how the results relate to the research questions stated in the introduction of this thesis will be discussed. Since RQ 2 and RQ 3 are sub-questions for answering RQ 1, these will be discussed first.

6.1 RQ 2: What are the challenges when using TracIMo in this context?

The challenges in using TracIMo can be divided in two groups. They were either related to eliciting traceability goals or to deriving the traceability process and selecting and customizing the needed tools.

6.1.1 Eliciting traceability goals

Eliciting traceability goals that would support the current development of infection control products, prepare the company for the future development of diagnostics tools as well as feasible to fulfill during the course of this study was a significant challenge. In the end, such traceability goals were identified, although this required many hours of interviews to be conducted and potential traceability goals from other development areas to be discarded. This case study illustrates that a company might be interested in traceability due to regulatory reasons but struggle to identify benefits from the developers point of view. There are several potential reasons why using TracIMo to elicit suitable traceability goals in this context was challenging.
One reason is that, aside from the long-term plan of sometime in the future going back to developing medical devices, 1928 Diagnostics' development team had not put much thought into how or why traceability could be beneficial for them. Instead of a bottom-up demand for traceability, it was mostly a top-down desire from the QA/RA director responsible for this project taking place. Consequently, the fact that the QA/RA director wasn't very involved in the development team's day-to-day work is probably one factor in explaining why identifying suitable traceability goals took time. The director did point at one area where traceability could currently be beneficial though, namely maintaining HIPAA compliance, which was an area where he had been involved with the development team. However, it was not seen as the most suitable development area for this thesis due to the lack of a mature requirement management system. It was also in the context of external requirements where the developers were most familiar with the benefits of traceability. As such, eliciting suitable traceability goals that would benefit them directly took time.

While interviewing the CTO and developers, inspiration was taken from an interview guide developed by the authors of TracIMo aimed at helping the interviewer understand the development process and traceability goals. Although the guide provided guidance on how to elicit detailed traceability goals once a use for trace links is already known, it did not provide much guidance on how to identify new development areas where the introduction of traceability would be useful. The only questions in the interview guide aimed at identifying new uses for traceability was *"Apart from the current use case, do you have other areas in your development where you need traceability?"*. An interview guide less focused on the details of how trace links should be created and used and more focused on how to identify development areas where traceability is suitable would have been more helpful. In hindsight, when asking about the artifacts used in a development process, asking questions such as *"Is it important to have easy access to this artifact? In what circumstances?"* and *"Are there problems in accessing these artifacts?"* early on would have been helpful. This is because although an artifact may be useful to create, if it is not important to improve the accessibility to this artifact, then creating trace links to this artifact is probably not cost-beneficial. Asking this kind of question early on would have helped the researcher filter away artifacts that were in no need of traceability.

A fundamental reason why eliciting traceability goals was difficult was that the development process, regardless of which development area, did not depend on requirements for planned changes to be documented, whether it be in the form tickets or any other form. This impeded traceability benefits that rely on requirements being documented from being achieved. What follows is an elaboration on what is meant by this and how it relates to product conformance, change impact analysis and project accountability.

Cleland-Huang [4] explains how agile teams can most easily benefit from traceability by creating trace links from test cases to requirements and thereby supporting product conformance. However, since tickets (where requirements could be documented) were not always created, introducing a traceability strategy that relied on tickets did not make sense. Nor was improving product conformance for changes in general something the company think they needed. Partially thanks to the small size of the team and amount of interaction between developers, things on that front were working fine.

Similarly, this impedes a traceability strategy aiming at improving change impact analysis from being introduced. Without being able to rely on tickets for planned changes to exist, it becomes difficult to create trace links to artifacts that would be affected by the change. Once again though, nor was improving change impact analysis something sought for by the development. As explained by the CTO, their view was that by striving for CI/CD, improving their ability to analyse the impact and estimate the effort of a planned change was not necessary [1].

[1] Incidentally, traceability goals related to change control of validation documents were at the end included, although some aspects of this goal were later deprioritized (scenarios 5.1 and 5.2). The remaining aspect (scenario 5.3) could due to the shortage of time not be evaluated, so it remains to be seen if change control could improve after all. Nonetheless, the remaining aspect was not related to improving change analysis or effort estimation, it was rather about improving how to handle making needed changes to a validation document. Therefore, future improvements in this area can't be used to argue against their view that CI/CD is sufficient for them.

Lastly regarding project accountability, i.e. ensuring that all changes are for specified requirement and not excess functionality, is impossible to achieve if all requirements aren't specified. Again, this was not a benefit they were looking for either though.

The lesson here is that although many agile development teams use tickets, including 1928 Diagnostics, it can't be assumed that these teams use them to an extent where relying on them for a traceability strategy is suitable. Significant time and effort could have been saved if this was earlier accepted.

A more general reason for why traceability goals were difficult to elicit is likely due to the little importance given to documentation in general, whether it be tickets, user stories, requirements or documents. As already mentioned, documentation was not important for the general development process and for the team to be productive. The development area in which a traceability strategy was implemented, i.e. the process of creating and validating cgMLST schemas, was one of the few identified development areas where documentation (e.g. benchmark results) had a very important role. Looking at the different kinds of traceability benefits presented by Appleton [21] [4]: product conformance, process conformance, change impact analysis, project accountability, baseline reproducibility and organizational learning, all of them involve some kind of documentation.

Looking again at the benefits presented by Appleton, most of the traceability benefits achieved during this project can be categorized as either baseline reprocudibility or process conformance improvements. By automatically storing artifacts and creating trace links between them, 1928 Diagnostics was able to improve their ability to reproduce cgMLST schemas and their benchmark results. Similarly, by storing benchmark results, any bioinformatician can verify that the results stated in a validation document are accurate.

An interesting observation is that while presenting and discussing Appleton's traceability benefit with the CTO, this development area was not discussed since the researcher was not aware about it at the time and the CTO did not bring it up. Eliciting traceability goals by first presenting documented traceability benefits was in other words not successful. This project can therefore not show a successful method to elicit new traceability goals that haven't been thought about before, other than by conducting many interviews to understand different development areas until such goals present themselves. A rule of thumb might be to identify development areas where documentation is not just occasionally used, but play an important role and are relied upon.

A final reason is likely due to the small size of the development team. According to Cleland-Huang [4], the value of traceability generally increases as the size of a project increases. This theory was also supported by some of the interviewed developers themselves. As the team and platform will continue growing, they could see tickets getting a more important role and traceability becoming more beneficial in more areas.

6.1.2 Deriving the traceability process and selecting and customizing tools

A significant difficulty in introducing the traceability strategy was that external tools where explicit trace links could be created and stored, such as Jira and Bitbucket where associated tickets and pull-requests can be traced, was not used. Instead, a traceability process using in-house tools had to be designed and developed. This led to the following challenges.

1. **Tool integration:** Due to the development process being divided in three steps (schema creation, benchmarking and validation document creation) and the goal of automatically creating the trace links during each step, it was necessary to use three separate tools, i.e. the pipeline for generating cgMLST schemas, the benchmarking script and the script for generating validation documents. The tools thus had to be designed meticulously to ensure that the tools supported each others. For example, the benchmarking script needed to return the benchmark results in a way that would make it easy for the script for generating validation documents to access and parse them. Tool integration is a known traceability challenge that has for example been addressed in a previous study [26] where an application lifecylce management (ALM) platform was integrated with testing tools. This study shows how in-house tools can also be integrated.
2. **Fragmentation of where artifacts could be stored:** Due to restrictions set by the development process, consolidating all relevant artifacts in a single location such as the main repository was not possible. Some artifacts thus had to be stored in other repositories or in S3. This fragmentation meant that a patchwork of different kinds of trace links had to be designed based on the features and restrictions of each location. Designing trace links between artifacts stored in different locations was particularly challenging. For example, tracing an artifact to the version of the script that generated it is easy if both are stored in a git repository, git inherently makes this possible. But if the generated artifact is stored elsewhere, this becomes more challenging. As these restrictions were further examined during the derivation of the traceability process, changes also had to be made to the TIM (since released cgMLST schemas had to be stored in the main repository and in S3).
3. **Trace link directionality:** When using external tools, making all trace links bidirectional is usually not an issue. For example, when a pull-request in Bitbucket is connected to a ticket in Jira, the user can see this connection from both artifacts and trace between them in both directions. Perhaps therefore, trace link directionality is not emphasized in TracIMo. In this project though, paying attention to directionality turned out to be very important. Some of the trace links created by the tools were only unidirectional. Although making all trace links bidirectional would have been possible, it would have taken more time and effort while not being necessary to fulfill the traceability goals. To save time and effort spent, ensuring that the TIM only specified the directionalities that were necessary for the traceability goals was very important.

TracIMo acknowledges the possibility for companies to use in-house tools but does not discuss the implications of taking this route. The methodology does however state that *"it is important to ensure that the selected tool can be customized in a reasonable time frame and cost"*.
This study exemplifies the kind of challenges that a company may expect when taking this route. The tools must be designed and developed rather than selected and customized. Although the introduced traceability strategy had various benefits, the challenges made designing and developing the tools a complicated and lengthy process. In addition, there is a high learning curve for others to understand the traceability process. Just to develop a prototype of the tools, the bioinformatician had to spend approximately one month.
Another thing worth mentioning is that even though this report presents the derivation of the traceability process and the selection of tools (or rather designing of tools) as separate steps, in reality these steps were conducted in parallel. This was done because deriving a traceability process naturally led to discussions in how the tools could be designed to fulfill the process. Other companies that wish to develop in-house tools may want to do the same.

6.2 RQ 3: What are the benefits and drawbacks of the designed traceability strategy in this context?

Even though the traceability strategy was not fully implemented, it is clear that it will support the bioinformaticians creating and validating cgMLST schemas. Most importantly, relevant artifacts will automatically be uploaded in a way that allows bioinformaticians to more easily access them. In addition, validation documents will become easier to create and more standardized. The traceability strategy also aimed to improve change control of the validation documents. Lastly, bioinformaticians will more easily be able to add benchmark results from new datasets to validation documents, although how to handle the manually created parts is a remaining challenge that must be addressed.
A drawback is that using documented commit hashes as trace links takes many steps and commit hashes won't work if a branch was merged or rebased incorrectly. Another drawback of the traceability strategy is that it forces bioinformaticians to properly implement a database before being able to use the benchmarking script on that database. This was also seen as a benefit though since increasing the amount of properly implemented datasets was a process goal. This restriction will motivate bioinformaticians to make the process of implementing databases easier.
The most significant cost in introducing the traceability strategy was that in-house tools had to be developed. Taking this route had various challenges and the tools will need to be further developed but the company expects the strategy to be cost-beneficial in the long-term. An important reason why this route was chosen was because the company did not want to make the process more complicated by introducing external tools and they wanted the trace links to be created automatically.

6.3 RQ 1: How can TracIMo be used to introduce a traceability strategy in an agile environment?

Even though there were challenges in introducing the traceability strategy, the results of this thesis show how TracIMo could be used to introduce one at 1928 Diagnostics. Introducing it was costly and it had drawbacks, but the company expects the cost-benefit to be positive in the long-term.
TracIMo proved to be helpful by providing a systematic methodology. Steps 1 to 5, where the traceability goals were elicited and systematically assessed and a measurement plan was made, were particularly helpful for the researcher. Once the traceability goals were clear and measurement plan was clear, the execution of subsequent steps, i.e. deriving the traceability process, selecting and customizing tools, deploying the traceability strategy and the evaluation, came more naturally. However, the methodology was still helpful during these steps to ensure that no step was forgotten.
The results also show that TracIMo could be used to help 1928 Diagnostics be more prepared for complying with medical device regulations. The development area where the traceability strategy was introduced was not regulated, but it was chosen knowing that a similar traceability strategy could be beneficial in a regulated environment. As the QA/RA director explained, if a process for managing requirements were to be added, e.g. by explicitly stating them in the validation documents, a similar traceability strategy could be beneficial for tracing requirements to their verification.

7

Validity threats

In this chapter, threats that could affect the validity of this study's findings are discussed. The threats are categorized and defined following Staron's book on action research [12], although an additional category taken from Runeson and Höst [27], reliability, has been added.

7.1 Construct Validity

Construct validity raises the question whether what the researcher intended to study matches what is actually studied.
A mismatch may occur if there are misunderstandings between the researcher and interviewees. In this study, various measures were taken to reduce the risk of misunderstandings. While eliciting traceability goals, member checks with the CTO, QA/RA director and interviewed bioinformatician were conducted. Regular discussions with the interviewed bioinformatician were held to resolve any misunderstanding about the development process, the derived traceability process and the developed tools.
Measures were also taken to avoid misunderstandings with the QA/RA director who did not have a deep understanding about the chosen development area or the developed tools. The researcher, CTO and interviewed bioinformatician explained enough for the QA/RA director to have a basic understanding. The researcher avoided eliciting details regarding traceability goals from the QA/RA director that were beyond his understanding, these details were elicited from the CTO and interviewed bioinformatician instead. During the focus group, the QA/RA director was not asked to evaluate the strategy's cost-benefit or the applicability in other development areas (questions 2 and 3 in the focus group). However, he did have enough of an understanding to evaluate how the traceability strategy could be beneficial in a regulated environment (question 4 in the focus group). The reason for this is because regulations don't specify how development artifacts should be traceable, only how the verification and requirements should be traceable, parts that he did understand.
While measuring the effects of an intervention, it is also important to make use of measurement method triangulation, i.e. using different methods to measure the same effect, to ensure that the same observation is made regardless of the measurement method. In action research, this can be done by having both quantitative and qualitative metrics. In this study, almost all metrics were qualitative. Including more quantitative metrics was considered but after having discussed it with the

interviewed bioinformatician, we unfortunately realized that it would not be feasible to collect enough quantitative data. There are various reasons for this that are discussed below. Not being able to make many quantitative measurement means that the certainty of the qualitative measurements and the findings of this study are reduced.
First of all, quantitatively measuring the accessibility of artifacts by for example counting the number of steps it took to access an artifact before the intervention was also not suitable since how the artifacts were to be kept wasn't standardized. The number of steps was dependent on who and where the artifact was stored (if it had been stored at all) and who was later trying to access it. Due to this reason, gathering enough data to calculate the average time it took for different bioinformaticians to access an artifact before the intervention would also have taken a prohibitive amount of time away from the bioinformaticians.
Collecting enough data on the ease of creating validation documents by measuring the amount of time it takes to create validation documents before and after would not be feasible since validation documents are not created very often and recreating documents with the old method would be take too much time away from the bioinformaticians. Quantitatively measuring how much of the validation document could automatically be generated was possible though (both standardized parts could be generated, although the parameters used in the pipeline for generating cgMLST schemas was still missing).
Another construct validity threat is that this action research study only follows one cycle, although almost all action research projects should have more [12]. If this thesis were to continue, the researcher would follow another cycle. The company will continue working on the implemented traceability strategy by starting another cycle and using what they learned from the first cycle, but this will be done without the researcher. This action research study can thus be seen as incomplete. The reason why the study didn't follow more cycles is because we thought the first cycle produced enough knowledge to answer the research questions. However, it is possible that another cycle would produce more answers for the research questions.

7.2 Internal validity

Internal validity is relevant whenever causal relations are examined, e.g. to question whether improvements were caused by the traceability strategy or other uncontrolled factors. Like any action research project aiming for a long-term collaboration, there are certain internal validity threats that we must pay special attention to: history effects, maturity and biased selection of subjects [12].
History effects are effects from events occurring before or between measurements. A historic effect that may have had an influence was the COVID-19 pandemic causing developers to work remotely and interviews being conducted on Zoom. It's possible that collaborating with the company face-to-face would have made eliciting traceability goals and introducing the traceability strategy less challenging. Although the discussed challenges perhaps would have been easier to deal with, we believe they would have been present in either case.
Maturity refers to how things naturally change by themselves as time progresses.

It's possible that improvements in the development process could be due to maturity rather than the traceability strategy. This is seen as unlikely though since the development processes was closely examined before introducing the traceability strategy and the only two bioinformaticians who use the development process were involved in the project. The bioinformaticians reported no changes to the development process other than the ones made by the traceability strategy.
Biased selection of subjects can occur from the temptation to focus on subjects with whom one has a good collaboration. For the chosen development area, it was only relevant to focus on two developers, i.e. the two bioinformaticians. Although more time was spent with the interviewed bioinformatician, i.e. the one who developed the tools, the researcher took input from both during the evaluation.

7.3 Conclusion Validity

Conclusion validity raises the question whether the correct conclusions have been drawn from the observations. Action research may introduce bias if people who were involved in the intervention are also asked to evaluate the improvements, i.e. drawing conclusions based on their observations. People are prone to judge their own work less objectively. This may be the case in this study, since the interviewed bioinformatician was very involved in introducing the traceability strategy and was then also part of the evaluation. To reduce the risk of any bias being introduced, another bioinformatician, the CTO and the QA/RA director were also part of the evaluation. There were no disagreements during the evaluation but the presence of bias can't be ruled out completely.

7.4 External validity

External validity raises the question to what extent the findings can be generalized. The particularities of the context, i.e. the company, makes it more difficult to generalize the findings. To help other practitioners evaluate to what degree the findings of this study may be generalized to other contexts, this study has aimed to be transparent and meticulous while presenting 1928 Diagnostics and its characteristics.

7.5 Reliability

Reliability raises the question whether the results were influenced by the researchers. For example, the fact that all data was collected by a single researcher, could be seen as a reliability threat. To address this validity threat, the questions used in the evaluation were added to the appendix. However, due to the discussed challenges in eliciting traceability goals and the complexity of the development processes, eliciting traceability goals by asking a predefined list of questions was not possible. Some questions were prepared before each interview but they changed from interview to interview and other questions were improvised as the researcher learned more and more about the development processes and goals. Therefore, interview questions

from the first phase of this study are not presented. Nonetheless, we recommend future researchers interested in deploying TracIMo to read the interview guide provided by TracIMo's authors, from which this study took inspiration [1].

[1] https://tinyurl.com/y3n96ldq

8

Conclusion and future work

In this action research study, a traceability strategy was introduced at 1928 Diagnostics, a start-up with an agile team of developers. This was done by applying TracIMo, a methodology for systematically introducing software traceability in companies.
First, interviews were conducted to understand the different development processes and elicit traceability goals. The traceability goals were defined by systematically analyzing, assessing and modelling the collected data and conducting member checks. Next, a traceability process was derived and needed tools were developed in collaboration with a developer at the company. Thereafter, a prototype of the tools were tested and evaluated by two developers. Lastly, a focus group with the two developers and the CTO and QA/RA director was also organized to evaluate more aspects of the traceability strategy.
This process was thoroughly reported in the results of this study. Introducing the traceability strategy had a number of costs, challenges and drawbacks, which have also been discussed in this study, but in the long-term the company expects it to be cost-beneficial. The benefits were also presented and could motivate more companies to introduce traceability. This study contributes by showing how a traceability strategy can be introduced in an agile environment and serving as a second evaluation of the TracIMo methodology.
Regulations were taken into consideration while introducing the traceability strategy, although the traceability strategy was not used in a regulated environment. To evaluate how TracIMo can be used to introduce a traceability strategy in a regulated environment, TracIMo must be used at a suitable company.

Bibliography

[1] Frisk, D. (2016) A Chalmers University of Technology Master's thesis template for LaTeX. Unpublished.

[2] Maro, S., Steghöfer, J.-P., Bozzelli, P., & Muccini, H. (2020) TracIMo: A Traceability Introduction Methodology and its Evaluation in an Agile Development Team. In Maro, S., Improving software traceability tools and processes (pp. 153-197). http://hdl.handle.net/2077/65837

[3] Golemshinska, K. & Kamsheh, R. (2019) Medical Device Software Organization, Department of Computer Science and Engineering, UNIVERSITY OF GOTHENBURG, CHALMERS UNIVERSITY OF TECHNOLOGY.

[4] Cleland-Huang, J., Gotel, O., & Zisman, A. (2012) Software and Systems Traceability

[5] Hanssen, G., Stålhane, T., & Myklebust, T. (2018) SafeScrum - Agile Development of Safety-Critical Software.

[6] Hoang Duc, V. (2015) Traceability in agile software projects. Master's thesis, Department of Computer Science and Engineering, University of Gothenburg.

[7] Mäder, P., & Egyed, A. (2014) Do developers benefit from requirements traceability when evolving and maintaining a software system?

[8] Mäder, P., Jones, P., Zhang, Y., & Cleland-Huang, J. (2013) Strategic traceability for safety-critical. IEEE Software, 30(3):58-66

[9] McCaffery, F., Casey, V., Sivakumar, M., Coleman, G., Donnely, P., & Burton, J. (2012) Medical Device Software Traceability. In Cleland-Huang, J., Gotel, O., & Zisman, A., Software and Systems Traceability (pp. 321-339).

[10] Regan, G., McCaffery, F., McDaid, K., & Flood, D. (2012) Traceability-Why Do It? International Conference on Software Process Improvement and Capability Determination, (pp. 161-172)

[11] Rempel, P., Mçder, P., & Kuschke, T. (2013) An empirical study on project-specific traceability strategies. 21st IEEE International Requirements Engineering Conference, (pp. 195-204).

[12] Staron, M. (2020) Action Research in Software Engineering.

[13] Health Insurance Portability and Accountability Act of 1996 (HIPAA), https://www.cdc.gov/phlp/publications/topic/hipaa.html

[14] Van Solingen R, Basili V, Caldiera G, Rombach HD (2002). Goal/Question/Metric (GQM) approach. Encyclopedia of Software Engineering

[15] Steghöfer, J.-P. (2017) Software traceability tools: Overview and categorisation. In: Report of the GI Working Group "Traceability/Evolution", German Infor-

matics Society (GI), pp 2-7, http://pi.informatik.uni-siegen.de/stt/38_1/01_Fachgruppenberichte/ARC_AKTE/ARC_AKTE_2017_p2_steghoefer.pdf
[16] 1928 Diagnostics' website, retrieved 15/07/2020 https://www.1928diagnostics.com/product/index.html
[17] Manifesto for Agile Software Development, Kent Beck,Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland & Dave Thomas (2001) https://agilemanifesto.org/
[18] Jira Software, https://www.atlassian.com/software/jira
[19] Bitbucket, https://bitbucket.org/
[20] Basecamp, https://basecamp.com/
[21] Appleton, B. (2005), Traceability and TRUST-ability, ACME Blog, http://bradapp.blogspot.com/2005/03/traceability-and-trust-ability.html
[22] Docker, https://www.docker.com/
[23] Amazon S3, https://aws.amazon.com/s3/
[24] Amazon Web Services (AWS) https://aws.amazon.com/
[25] RefSeq: NCBI Reference Sequence Database, https://www.ncbi.nlm.nih.gov/refseq/
[26] Amalfitano, D., De Simone, V., Fasolino, A.R., & Scala, S. (2017) Improving traceability management through tool integration: an experience in the automotive domain. In Proceedings of the 2017 International Conference on Software and System Process (ICSSP 2017). Association for Computing Machinery, New York, NY, USA, 5–14. https://doi.org/10.1145/3084100.3084101
[27] Runeson, P., Höst, M. Guidelines for conducting and reporting case study research in software engineering. Empir Software Eng 14, 131 (2009). https://doi-org.proxy.lib.chalmers.se/10.1007/s10664-008-9102-8

A

Appendix 1

A.1 Group Interview Questions

Table A.1: Interview questions used during group interview with bioinformaticians mapped against traceability goals and metrics. Traceability goals have been shortened to reduce clutter.

TG 1:	Improve accessibility from a validation document to the exact versions of artifacts that went into the generation and validation of the cgMLST schema.
Metric 1.1	Is it now easier to locate the ... from a validation document? Why/how? Do you think it will be different for validation documents for older released cgMLST schemas? **Q 1.1.1:** ...released cgMLST schema... **Q 1.1.2:** ...benchmark results... **Q 1.1.3:** Is it easy to locate the script for generating validation documents from a validation document? Why/how?
Metric 1.2	How much effort is it to use the script for generating validation documents to create trace links from a validation document to ... ? How could the process and tool improve? **Q 1.2.1:** ...the released cgMLST schema (by uploading the validation document and schema to the main repository in a modular structure)... **Q 1.2.2:** ...the corresponding cgMLST schema in S3 (by renaming it to match the released cgMLST schema's name)... (*Cost evaluation of creating trace links from the cgMLST schema in S3 to the benchmark results is covered by question 3.2.2.)* **Q 1.2.3:** ...the script for generating validation documents (by marking the document with the commit hash)...
TG 2:	Improve accessibility from a cgMLST schema to the exact versions of artifacts that went into the generation of the schema.

Table A.1: Interview questions used during group interview with bioinformaticians mapped against traceability goals and metrics. Traceability goals have been shortened to reduce clutter. *(Continuation)*

Metric 2.1	Is it now easier to locate ... from a cgMLST schema? How/why? Do you think there is a difference if it is from *a)* the latest released cgMLST schema, *b)* an older released cgMLST schema and *c)* a preceding unreleased cgMLST schema? **Q 2.1.1:** ...the pipeline for generating cgMLST schemas version... **Q 2.1.2:** ...the parameters used in the pipeline for generating cgMLST schemas... **Q 2.1.3:** ...the reference genomes... **Q 2.1.4:** ...preceding cgMLST schemas...
Metric 2.2	How much effort is it to use the pipeline for generating cgMLST schemas to create trace links from a cgMLST schema to ... ? How could the process and tool improve? **Q 2.2.1:** ...the pipeline for generating cgMLST schemas version (by documenting the commit hash in an accompanying file)... **Q 2.2.2:** ...the parameters used in the pipeline for generating cgMLST schemas (by documenting them in an accompanying file)... **Q 2.2.3:** ...the reference genomes (by documenting them in an accompanying file)... **Q 2.2.4:** ...preceding cgMLST schemas (by storing all cgMLST schemas in a modular structure)...
TG 3:	Improve accessibility from benchmark results to the exact versions of artifacts used to produce the benchmark results.
Metric 3.1	Is it now easier to locate ... from the benchmark results? How/why? **Q 3.1.1:** ...the dataset... **Q 3.1.2:** ...the benchmarking script and cgMLST analysis pipeline... **Q 3.1.3:** ...the parameters used in the cgMLST analysis pipeline... **Q 3.1.4:** ...the cgMLST schema...

Table A.1: Interview questions used during group interview with bioinformaticians mapped against traceability goals and metrics. Traceability goals have been shortened to reduce clutter. *(Continuation)*

Metric 3.2	How much effort is it to use the benchmarking script to create trace links from benchmark results to ... ? How could the process and tool improve? **Q 3.2.1:** ...the dataset(s) (by documenting them in an accompanying file)... **Q 3.2.2:** ...the benchmarking script and cgMLST analysis pipeline (by documenting the commit hash in an accompanying file)... **Q 3.2.3:** ...the parameters used in the cgMLST analysis pipeline (by documenting them in an accompanying file)... **Q 3.2.4:** ...the cgMLST schema (by uploading benchmark results to S3 in a modular together)...
TG 4:	Improve creation process of validation documents by allowing standardized parts to be automatically generated with the help of a script and trace links to related artifacts.
Metric 4.1	**Q 4.1:** Are the validation documents of higher quality? How/why?
Metric 4.2	**Q 4.2:** Are the validation documents more standardized? How/why?
Metric 4.3.1	**Q 4.3.1:** Is it now easier to create validation documents? How/why?
Metric 4.3.2 *(Quantitative)*	**Q 4.3.2:** How many standardized parts are now automatically generated?
TG 5:	Improve change control of validation documents with the help of trace links.
Metric 5.1	**Q 5.1:** If the script for generating validation documents is changed, is it easy to decide which validation documents should be updated? How/why?
Metric 5.2	**Q 5.2:** If a validation document needs to be updated due to the script for generating validation documents having changed, is it easy to do so? How/why?
Metric 5.3	**Q 5.3:** If a new dataset for an already released cgMLST schema is properly implemented and benchmarked, is it easier to update the validation document accordingly? How/why?
TG 1, 2, 4:	Traceability goals that are fulfilled differently for already released cgMLST schemas. (I.e. when trace links are create retroactively)
Metric 1.2	**Q 7.1:** How much effort is it to create a trace link from a new validation document to an old validation document? (By uploading a copy of the old validation document in Google Drive to the main repository)

Table A.1: Interview questions used during group interview with bioinformaticians mapped against traceability goals and metrics. Traceability goals have been shortened to reduce clutter. *(Continuation)*

Metric 2.2	**Q 7.2:** How much effort is it to find the reference genomes that were used to create an already released cgMLST schema? **Q 7.3:** How much effort is it to find the parameters that were used in the pipeline for generating cgMLST when creating an already released cgMLST schema?
Metric 4.1, 4.2, 4.3.1	**Q 7.3:** How would you answer questions 4.1, 4,2, 4.3.1 when creating validation documents for already released cgMLST schemas? Is there any significant difference?
TG 1-3:	Traceability goals with metrics regarding the cost of maintaining trace links.
Metric 1.2, 2.2, 3.2	**Q 8:** Do you foresee any extra efforts needed to maintain accurate trace links? E.g. if an artifact is renamed or deleted. If so, for which trace links, how much effort and how often?

A.2 Group interview answers for each metric.

TG 1:	Improve accessibility from a validation document to the exact cgMLST schema, benchmark results and script for generating validation documents *(see Goal 4 for explanation of this script)* that were used to generate the validation document. This does not only apply to the latest released version of a cgMLST schemas, but also to older released versions.
Question 1:	Is it easier to access the mentioned artifacts?
Metric 1.1:	(B) Opinions and perceptions on the accessibility from a validation document to the mentioned artifacts, with and without trace links.
Metric 1.2:	(C) Opinions and perceptions on the cost of using the process and tools for creating and maintaining the trace links from a validation document to the mentioned artifacts.

Evaluation of metric 1.1:

- Accessibility to cgMLST schema:
  - Validation documents can be generated but they are not yet uploaded to main repository, meaning this trace link is still missing. The name of validation documents is also missing the schema version, making it difficult to trace to the corresponding cgMLST schema whenever there are several released versions. Fixing these things is easy though and will soon be accomplished.
  - Once these things are fixed, there will be a noticeable improvement in accessibility from validation documents to cgMLST schemas since they will be much closer to each other.

- Accessibility to benchmark results:
  - There is a big improvement in being able to access benchmark results that have been generated with the new script. Now, once the bioinformatician has located the corresponding cgMLST schema in S3, finding the benchmark results is easy.
- Accessibility to script for generating validation documents:
  - Bioinformaticians agree that accessing the script is easy by simply looking at the commit of the validation document.

Evaluation of metric 1.2:

- Scripts make creating trace links very low / no effort.
- However, at the moment the benchmarking script is not optimized and more likely to fail or crash, meaning that extra effort is needed to create and maintain trace links.

TG 2:	Improve accessibility from a cgMLST schema to the exact versions of artifacts that went into the generation of the schema from a bioinformatician's point of view, i.e. pipeline for generating cgMLST schema (including parameters used), reference genomes and preceding cgMLST schemas. This does not only apply to the latest released version of a cgMLST schema, but also to older released/unreleased versions.
Question 2:	Is it easier to access the mentioned artifacts?
Metric 2.1:	(B) Opinions and perceptions on the accessibility from a cgMLST schema to the mentioned artifacts, with and without trace links.
Metric 2.2:	(C) Opinions and perceptions on the cost of using the process and tools for creating and maintaining trace links from a cgMLST schema to the mentioned artifacts.

Evaluation of metric 2.1:

- Accessibility to pipeline for generating cgMLST schema:
  - Moving forward, it will for the first time be possible to know which version of the pipeline was used to create new schemas thanks to the commit hash being kept within schema.
- Accessibility to parameters for pipeline for generating cgMLST schema:
  - They are now automatically saved in a standardized way, which is an improvement.
  - However, it would be useful if parameters that have deviated from the default are highlighted. Current solution of finding parameters within config file is a bit slow. Once parameters get parsed and added to the validation document, this will be less of an issue though (still an issue for schemas that don't get released though).
- Accessibility to reference genomes:
  - They are now automatically saved in a standardized way, which is an improvement.
  - For some cgMLST schemas, the list of reference genomes wasn't kept. Retroactively creating trace links for these schemas will not be possible.

- Accessibility to preceding cgMLST schemas:
  - Substantial improvement. From not being able to find or recreate them, where to keep them is now standardized and they are accessible for everyone.
- Retroactively creating trace links for already released cgMLST schemas:
  - Should in theory be possible.
  - In some cases, parameters and reference genomes have not been kept. Creating trace links for these schemas won't be possible.

Evaluation of metric 2.2:

- From manually having to save the artifacts, the script now does it for you while at the same time creating the trace links with very low effort.

TG 3:	Improve accessibility from benchmark results to the exact versions of artifacts used to produce the benchmark results from a bioinformatician's point of view, i.e. the cgMLST schema, dataset, benchmarking script and cgMLST analysis pipeline (including parameters used). This does not only apply to benchmark results of the latest released version of a cgMLST schema, but also for older released/unreleased versions.
Question 3:	Is it easier to access the mentioned artifacts?
Metric 3.1:	(B) Opinions and perceptions on the accessibility from benchmark results to the mentioned artifacts with and without trace links.
Metric 3.2:	(C) Opinions and perceptions on the cost of using the process and tools for creating and maintaining trace links from benchmark results to the mentioned artifacts.

Evaluation of metric 3.1:

- Accessibility to datasets:
  - Biggest difference is that it is now enforced for datasets to be properly implemented, since the benchmarking script does not work otherwise.
  - Accessing datasets that have properly been implemented is very easy (both before and now), so by enforcing datasets to be properly implemented, datasets can always be accessed easily. Datasets that had not been properly implemented were less accessible.
  - Effort: Properly implementing datasets takes time and was before often avoided, so now that it is enforced, there will be an increase in effort needed before being able to benchmark. This process can be optimized and increasingly automatized though.
- Accessibility to benchmarking script:
  - Instead of hard-coding and using different benchmarking scripts which aren't kept for each schema and dataset, a single benchmarking script is now used.
  - Knowing which version of the script was used is possible thanks to the commit hash being kept within the results, although then accessing the script version is a bit clunky.

- Accessibility to cgMLST analysis pipeline:

missing

- Accessibility to cgMLST analysis pipeline parameters:
  - Parameters are now saved automatically in a standardized way, which is an improvement.
  - File containing parameters is not parsed though and one has manually see if parameters deviate from the default.

Evaluation of metric 3.2:

-

TG 4:	Improve creation process of validation documents from a bioinformaticians point of view by allowing standardized parts to be automatically generated with the help of a script and trace links to related artifacts.
Question 4.1:	Are the validation documents of higher quality?
Metric 4.1:	(B) Opinions and perceptions on the quality of validation documents, with and without trace links and the script.
Question 4.2:	Are the validation documents more standardized?
Metric 4.2:	(B) Opinions and perceptions on the standardization of validation documents, with and without trace links and the script.
Question 4.3:	Is it easier to create validation documents?
Metric 4.3.1:	(B) Opinions and perceptions on the ease of creating validation documents, with and without trace links and the script.
Metric 4.3.2:	(B) Number of standardized parts that are now automatically generated.
Note:	*Cost metrics for creating the trace links needed for this traceability goal is covered by the metrics for question 1, 2 and 3.*

The script for generating validation documents and releasing cgMLST schemas is in a very early state and won't be used in practice until it has been further developed.

Evaluation of metric 4.1:

- Risk of human errors is expected to decrease, which will help to improve quality.
- The generated validation documents are at the moment .txt files (for the sake of simplicity). This format is more limited than Google Docs, which affects the documents' level of quality. Another format that allows for more formatting, e.g. Markdown, shall be used in the future instead to increase the quality.
- The quality of standardized parts in terms of presentation and amount of content is currently inferior compared to the ones in manually created validation documents. Will have to see if the script can match a human in the future.

Evaluation of metric 4.2:

- Standardized parts that are automatically generated will make the documents more standardized.

Evaluation of metric 4.3.1:

- Not having to create tables, calculate means and medians etc. for the overview of the benchmark results makes it easier to create validation documents.

- How to handle the validation document being partially generated and partially manually created still needs to be worked out.
- Option to generate a validation document without uploading the cgMLST schema to the main repository is still missing. This option needs to be added if new validation documents are to be created for already released schemas.

Evaluation of metric 4.3.2:

- So far only one standardized part can be generated, the overview of benchmark results.
- Generating an overview of how the schema was generated, including information about the parameters used in the pipeline and the reference genomes, is expected to be added in the near future.
- manual part

TG 5:	Improve change control of validation documents with the help of trace links from a bioinformatician's point of view.
Question 5.1:	If the script for generating validation documents is changed, is it easy to decide which validation documents should be updated?
Metric 5.1:	(B) Opinions and perceptions on the ease of deciding which validation documents should be updated if the script for generating validation documents has been changed.
Question 5.2:	If a validation document needs to be updated due to the script for generating validation documents having changed, is it easy to do so?
Metric 5.2:	(B) Opinions and perceptions on the ease of updating a validation document with a new script version.
Question 5.3:	If a new dataset for an already released cgMLST schema is properly implemented and benchmarked, is it easier to update the validation document accordingly?
Metric 5.3:	(B) Opinions and perceptions on the ease of updating a validation document with a new dataset and associated benchmark results, with and without trace links and the script.
Note:	*Cost metrics for creating the trace links needed for this traceability goal is covered by the metrics for questions 1 and 3.*

Due to reasons mentioned in section 5.7.1, metrics 5.1 and 5.2 are not evaluated.

- Evaluation of metric 5.3:
  - Yes it is easier, although how to handle manually created parts still is still unknown.
  - This question will be addressed in the future when the script for generating validation documents has been used to create a real validation document.

A.3 Focus group questions

Table A.2: Questions used during focus group.

Q 1:	Do you have any more questions or feedback on the processes and tools?
Q 2:	How would you compare the cost of implementing and using the traceability strategy to the expected benefits? Are there parts of the traceability strategy that are more/less cost-beneficial?
Q 3:	Do you think there are other development areas where it would be worth to implement a similar traceability strategy? Which/why?
Q 4:	Do you think the current traceability strategy could be relevant in a regulated environment? What do you think is missing for the traceability strategy to be more relevant in a regulated environment?