Specification, Estimation and Monitoring of Quality-Related Software Strategic Indicators in Agile Software Development

Martí Manzano Aguilar

Supervised by:

Claudia Ayala

Cristina Gómez

PhD in Computing program

UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Disclaimer

This thesis has been submitted for assessment in partial fulfillment of the PhD in Computing program. The thesis is the result of my own work and includes nothing that is the outcome of work done in collaboration except where specifically indicated. Many of the ideas in this thesis were the product of discussion with my supervisors.

Excerpts of this thesis have been published or submitted as conference manuscripts and academic publications. Such publications are listed in Chapter 1.

Abstract

In line with the current trend of exploiting corporative data, software companies, especially those using Agile and Rapid software development, are challenged to improve the quality of their products, their profitability and efficiency by exploiting the large amount of data related to their software processes and products from the use of their corporate tools (e.g., continuous inspection tools, continuous integration tools, project management tools, and issue trackers).

Although such data exploitation has shown to be beneficial for supporting decision-making processes, the evidence shows that existing support is mostly related to operational decisions, letting aside the support for strategic decision making. Operational decisions are simple routine decisions linked to the effective and efficient execution of the daily operations within the company (e.g., test specification and implementation, bug tracking, version control, etc...). Strategic decisions refer to complex, non-routine decisions related to business goals and objectives.

The main problems that endanger the task of supporting strategic decision making through data exploitation are: a) the lack of approaches that help software companies to specify their own software strategic indicators (SSI). SSIs refer to measurable aspects (e.g., software quality, on-time delivery) that a software company considers important for their strategic decision-making processes, b) the inherent complexity of estimating SSIs, and c) the need of supporting the operationalization of the specification and estimation of SSIs by enabling their monitoring.

This PhD thesis aims to overcome these problems by:

- Devising a novel method called SESSI (Specification and Estimation of Software Strategic Indicators) that provides support for operationalizing the specification, estimation, and monitoring of SSIs in software companies. The method was conceived under design science and action-research principles in the context of the industrial partners of the Q-Rapids European project and applied to quality-related SSIs.

- Presenting how the use of the SESSI method and associated software supporting artifacts has shown promising results to enable an SSI monitoring infrastructure according to the needs and resources of a software company.

Additionally, this thesis explores the potential use of the resulting monitoring infrastructure and other related outputs from the SESSI method for enabling advanced decision-making support. In particular, a solution for forecasting the values of SSIs based on the SESSI method was applied in a software development company with positive results.

The results of this thesis aim to advance the state of the art on approaches to support evidence-based strategic decision making, in software companies using agile and rapid software development. The developed software support artifacts have been released as open source and can be reused and/or adapted by other software companies or researchers.

Resum

En línia amb la tendència actual d'explotació de dades corporatives, les empreses *software*, especialment les que utilitzen el desenvolupament *software* àgil i ràpid, tenen el repte d'aconseguir millores sobre la qualitat dels seus productes, així com la seva rendibilitat i eficiència mitjançant l'explotació de la gran quantitat de dades relacionades amb els seus processos i productes *software* provinents de les seves eines corporatives (per exemple, eines d'inspecció contínua, eines d'integració contínua, eines de gestió de projectes i eines de gestió d'errors).

Tot i que aquesta explotació de dades ha demostrat ser beneficiosa per donar suport als processos de presa de decisions, l'evidència mostra que el suport existent està principalment relacionat amb les decisions operatives, deixant de banda el suport per a la presa de decisions estratègiques. Les decisions operatives són simples decisions rutinàries vinculades a l'execució eficaç i eficient de les operacions diàries dins de l'empresa (per exemple, especificació i implementació de proves, seguiment d'errors, control de versions, etc.). Les decisions estratègiques es refereixen a decisions complexes i no rutinàries relacionades amb les metes i objectius empresarials.

Els principals problemes que dificulten la tasca de suport a la presa de decisions estratègiques mitjançant l'explotació de dades són: a) la manca de propostes que donin suport a les empreses *software* a especificar els seus propis indicadors estratègics *software* (SSI). Els SSI fan referència a aspectes mesurables (per exemple, qualitat del *software*, lliurament puntual) que una empresa *software* considera importants per als seus processos de presa de decisions estratègiques, b) la complexitat inherent de l'estimació dels SSI, i c) la necessitat de donar suport a la operacionalització de l'especificació i l'estimació dels SSIs per tal d'habilitar el seu monitoratge.

Aquesta tesi doctoral pretén superar aquests problemes mitjançant:

- El disseny d'un nou mètode anomenat SESSI (Especificació i Estimació d'Indicadors Estratègics *Software*) que ofereix suport per l'especificació, avaluació i seguiment dels SSI a empreses *software*. El mètode va ser concebut sota els principis de *design-science* i *action-research* en el context dels socis industrials del projecte europeu Q-Rapids i aplicat sobre SSIs relacionats amb la qualitat.

- La presentació de com l'ús del mètode SESSI i els artefactes de suport *software* associats han mostrat resultats prometedors per habilitar una infraestructura de monitoratge de SSIs d'acord amb les necessitats i recursos d'una empresa *software*.

Addicionalment, aquesta tesi explora l'ús potencial de la infraestructura de monitoratge resultant i altres sortides relacionades del mètode SESSI per donar suport avançat a la presa de decisions. Específicament, es va disenyar i aplicar una solució per predir els valors dels SSI basats en el mètode SESSI en una empresa *software* amb resultats positius.

Els resultats d'aquesta tesi tenen com a objectiu avançar l'estat de l'art quant a les solucions per donar suport a la presa de decisions estratègiques basades en evidències, en empreses *software* que utilitzen desenvolupament àgil i ràpid. Els artefactes *software* desenvolupats han estat alliberats com a codi obert i poden ser reutilitzats i/o adaptats per altres companyies *software* o investigadors.

Acknowledgements

I would like to express my sincere gratitude to my advisors Claudia Ayala and Cristina Gómez for their invaluable guidance and support throughout this journey. This thesis would have not been possible without them. I would also like to thank Xavier Franch for giving me the opportunity to start this thesis and for his constructive feedback.

I am also thankful to the GESSI group (including former members), especially Lidia L., Marc O., Cristina P. and Silverio M., for making me feel welcomed and integrated from the very beginning. During the thesis, I have had the opportunity to collaborate with talented researchers and professionals, including the ones from the Q-Rapids project. I am grateful for their effort and for accepting our fruitful collaborations. In this line, I want to notably thank Emilia Mendes and Antonin Abherve.

On a personal note, I thank my family, especially Ana, my fathers, my brother, and my in-laws for their unconditional love, understanding and moral support. I also thank my friends and my former roommates from the Hospitalet's flat.

This work has been possible thanks to the funding of the Q-Rapids project, grant agreement no. 732253 (Horizon 2020 program of the European Commission), the 2020 FI SDUR grant no. 00279 (Agència de Gestió d'Ajuts Universitaris i de Recerca, Generalitat de Catalunya, the GENESIS project, grant agreement TIN2016-79269-R (Spanish Ministry of Economy and Competitiveness), and the DOGO4ML project, grant agreement PID2020-117191RB-I00/AEI/10.13039/501100011033 (Spanish Ministerio de Ciencia e Innovación).

Contents

List of Figures

List of Tables

1 Introduction

1.1 Context

Nowadays, organizations are challenged to improve their profitability and efficiency by exploiting their corporate data (Janssen et al., 2017; Olszak, 2016). This is particularly true for the case of software-development intensive organizations. We define software-development intensive organizations (from now on, software companies) as public or private organizations extensively developing software either for third-parties, or for internal use.

Software companies produce large amounts of data related to their software processes and products from the use of their corporate tools (e.g., continuous inspection tools, continuous integration tools, project management tools, and issue trackers). Such data resides in corporate repositories. The exploitation of such data is considered key to improve their decision-making processes and gaining competitive advantage (Martínez-Fernández et al., 2018). Thus, the adoption of Data-Driven Decision Making (DDDM) approaches (i.e., the practice of basing decisions on the analysis of data rather than purely on intuition) has impacted greatly on software engineering research and practice over the last years (Figalist et al., 2021). The DDDM process comprises several general steps (see Figure 1) (Gill et al., 2014; Mandinach et al., 2006):

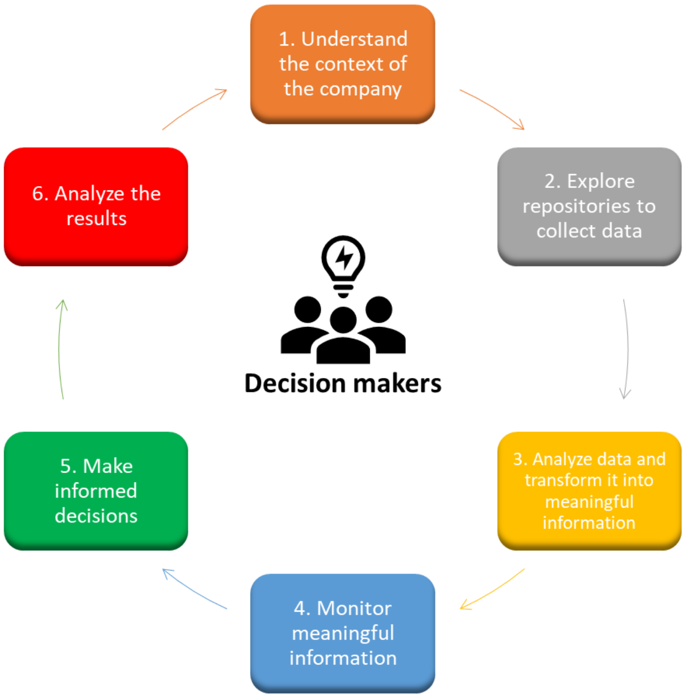

1. Understand the context of the company
2. Explore repositories to collect data
3. Analyze data and transform it into meaningful information
4. Monitor meaningful information
5. Make informed decisions
6. Analyze the results
Decision makers

Figure 1 Steps of a DDDM process

1) *Understand the context of the company*. Each company has its own resources and data, its particular constraints, goals and strategic plans. Analyzing such particularities is a crucial aspect for understanding the decision-making needs of each company.

2) *Explore the company's repositories to collect relevant data*. Once the context and needs of the company have been understood, further exploration should be performed to identify relevant data (usually from the available corporate repositories) to support the specific decision-making needs of the company. This includes the determination of how to automate the collection of such relevant data, usually through software tools.

3) *Analyze the collected data and transform it into meaningful information*. In order to support decision making, the collected raw data should be transformed into meaningful information that provides valuable insights to decision makers. Such meaningful information is usually specified as indicators. Indicators are defined in (IEEE, 2017) as "measures that provide estimates or evaluations of specific attributes with respect to defined information needs". To be actionable, the specification of indicators in a company must be aligned with the available data and an estimation procedure should be elaborated to compute the values of such indicators.

4) *Monitor meaningful information*. The systematic observation of meaningful information (e.g., indicators) in a visual way over a period of time (i.e., monitoring)

allows companies to perform further analysis of the behaviour, evolution and progress of relevant indicators.

5) *Make informed decisions*. Based on the monitored information, decisions makers may evaluate different alternatives and feel more confident to make the right decision.

6) *Analyze the results*. When decisions makers take a decision, they should evaluate the failure or success of the decision. In case of failure, monitoring the same insights discovered in the step 3 may help decision makers to understand the reasons of the failure and learn for the next loop.

Evidence exists that a great deal of software companies use indicators as essential assets for specifying and assessing meaningful information (Figalist et al., 2021; Zhang et al., 2013). Furthermore, software companies face a variety of decisions to make. These decisions typically fall into three categories, depending on the level at which they occur: operational, tactical and strategic decisions (Aurum et al., 2006; Matthies and Hesse, 2019; Moe et al., 2012). Strategic decisions are complex decisions and refer to business goals and objectives (i.e., budget aspects, product, and release plans) (Matthies and Hesse, 2019). Tactical decisions are less complex decisions and refer to the implementation of strategic decisions (i.e., identification and allocation of resources, and project management aspects in general) (Matthies and Hesse, 2019). Operational decisions are simple and routine decisions and refer to the effective and efficient execution of the daily operations within the company (e.g., test specification and implementation, bug tracking, version control, etc.) (Moe et al., 2012).

There are diverse software tools for supporting the management of tasks related to operational and tactical decisions. For instance, software development tools such as integrated developments frameworks, project or backlog management tools such as Redmine[1] or JIRA[2], continuous integration tools, bug tracker systems or software code assessment tools (e.g., SonarQube[3]) produce and collect valuable data related to operational and tactical decisions and provide indicators on diverse relevant aspects of

[1] https://www.redmine.org

[2] https://www.atlassian.com/es/software/jira

[3] https://www.sonarqube.org

the software development process (e.g., bug density, code complexity, compilation results, time tracking metrics, automatic Gantt and burndown charts generation and visualizations) (Figalist et al., 2022).

Strategic decisions can be supported by Business Intelligence (BI) tools. BI tools are used in different types of organizations for collecting and processing information, supporting decision makers with reports of indicators and monitoring dashboards to accelerate and improve their decisions (e.g., Tableau[4], Power BI[5]). The indicators and functionalities provided by these tools are not defined to cover the specific necessities of each software company (i.e., the data exploitation capabilities of BI tools do not precisely match with the usual tools used in the daily operation of software companies) nor the particular informational needs of a software company (Martínez-Fernández et al., 2018; Moe et al., 2012). This is, existing practical support for exploiting software companies' data for defining, estimating and monitoring their own indicators related to strategic decisions for their particular needs is scarce. Hereafter, we refer to such indicators as Software Strategic Indicators (SSI). SSIs refer to measurable aspects (e.g., software quality, on-time delivery) that a software company considers important for their strategic decision-making processes.

From the academia point of view, the situation is similar. Although there is considerable research on software metrics and indicators (López et al., 2022; Meidan et al., 2018), most proposals focus on indicators related to operational and tactical decisions, letting aside the problems related to provide SSIs. Whilst providing indicators related to operational and tactical decisions can be dealt by extracting and directly processing operational information from corporate repositories, providing suitable SSIs for a specific software company is endangered by several issues: the amount of heterogenous data needed (usually longitudinal data and sometimes with missing data periods) to define and estimate SSIs for informing such decisions, and the inherent complexity of strategic decision making. In addition, the fact that strategic decisions are practically influenced by tacit and explicit corporate knowledge (e.g., previous experiences, opinions and intuitions) (Svensson et al., 2019) has not been properly addressed in a practical solution (Moe et al., 2012).

[4] https://www.tableau.com

[5] https://powerbi.microsoft.com

1.2 Problem

As we have seen in the previous section, the use of DDDM approaches is a challenge for software companies. While this challenge exists regardless of the approach used to develop software, this thesis focuses on software companies using agile and rapid software development (ASD) which entail incremental and iterative software development methods guided by the agile manifesto (Beck et al., 2001). The main reason is that the industrial use cases approached in this thesis come from a European Project composed of industrial partners using ASD. Hereafter, when referring to software companies, we will assume those that use ASD. In addition, the industrial partners of the project were particularly interested on approaching SSIs related to quality.

At these respects, it should be remarked that: On the one hand, ASD methods (e.g., Scrum, and eXtreme Programming (XP)) are widely adopted throughout the software industry. Indeed, studies show a steady increase in the number of organizations adopting agile practices and processes over the last two decades, with an overwhelming popularity (Edison et al., 2022), corresponding to the 94% of adoption over the surveyed organizations in last published State of Agile Report (Digital.ai, 2021). On the other hand, market prospects indicate that up to 26% of firms' IT budgets are dedicated to software quality assurance and testing, and they predict an increase to 33% in the next three years (Buenen and Walgude, 2018).

In this context, the main problems tackled in this thesis are:

- **P1**. Need of approaches that help software companies to define (from now on, specify) their own SSIs (i.e., covering the particular needs of the company and its context) from the exploitation of their corporate data. Most of the existing proposals from research and practice (i.e., existing software tools) mainly deal with the specification of indicators related to operational and tactical decisions (Antinyan et al., 2014; Monteiro and De Oliveira, 2011; Padmini et al., 2015) and tackle the specification of those indicators in an ad-hoc manner, making hard to apply such proposals to different contexts and companies (Padmini et al., 2015; Perkusich et al., 2015). This is, the actual support for specifying SSIs that covers the particular needs of a company is scarce (Figalist et al., 2019;

Martinez-Fernandez et al., 2019). Dealing with this problem is not trivial and several challenges should be considered: a) the increasing amount of heterogeneous data generated by software companies that has to be explored to specify suitable SSIs and b) the complex nature of strategic decisions that increases the complexity of SSIs specification.

P2. Complex SSI estimation. Despite the importance of supporting strategic decisions in software companies (Dam et al., 2018; Figalist et al., 2021), support for estimating SSIs has not been sufficiently addressed in the literature and the industrial practice (Cito, 2016; Figalist et al., 2021; Martinez-Fernandez et al., 2019; Mesquida Calafat et al., 2022; Moe et al., 2012). Unlike indicators related to operational decisions, the estimation of SSIs is more complex and is endangered by several issues: a) one should deal not only with the huge amounts of heterogenous data but also with the fact that such data is longitudinal and sometimes contains missing data periods (given typical roll-backs and stages of the software development process) that highly jeopardizes data interpretation, b) the non-deterministic and subjective nature of the decision-making process (especially at the strategic decision level), requires that the SSI estimation considers not only data but also expert knowledge (Matthies and Hesse, 2019) and c) the complex nature of strategic decisions requires that SSIs are endowed with explainability features that enable decision-makers to understand the SSIs estimations. All in all, one should reconcile all these aspects to provide effective SSIs estimation support.

- **P3**. Need of approaches that support software companies to put forward the specification and estimation of their own SSIs for automatic monitoring purposes. The few literature about SSIs (see Chapter 2, section 2.2), do not provide further details for making them actionable. Some BI tools that offer data exploitation capabilities, hardly fit with the specific needs of companies about specifying and estimating SSIs. Guiding software companies to put forward an appropriate infrastructure for automatic monitoring of SSIs for supporting their decision-making processes is essential to promote industrial uptake.

Therefore, new approaches that help software companies to specify, estimate and monitor their SSIs from the exploitation of expert knowledge and corporate data are needed to effectively improve strategic decision-making processes in these companies (Martinez-Fernandez et al., 2019).

1.3 Objectives and research questions

In line with the problems detailed in the previous section, this thesis provides support to software companies (using ASD) to effectively exploit corporate repositories of heterogeneous data and expert knowledge for specifying, estimating, and monitoring their own SSIs and facilitating, in this way, their strategic decision making. Moreover, this dissertation provides guidance and software support to build the required infrastructure for putting forward such SSIs specification and estimation in order to enable SSIs automatic monitoring.

The scope of this thesis with respect to the DDDM cycle is depicted in Figure 2.

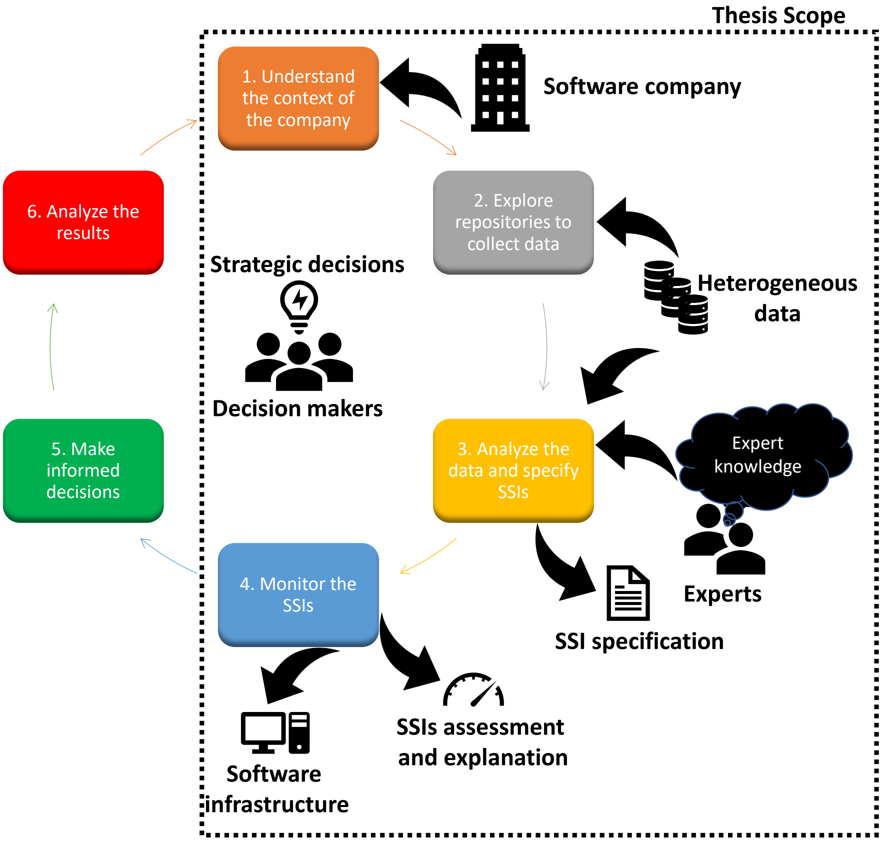


Figure 2 Thesis Scope

Thus, the general objective of this work can be stated as:

“To improve decision-making processes in software companies using ASD by providing support for specifying, estimating and monitoring SSIs from exploiting corporate repositories and expert knowledge in order to promote evidence-based decision making”

This high-level objective is further decomposed into the following objectives:

> **O1.** To support the specification of SSIs from exploiting corporate repositories and expert knowledge according to the decision-making needs and data availability.
>
> **O2.** To support the estimation of SSIs from exploiting corporate repositories and expert knowledge considering subjectivity, potential lack of data, and explainability needs related to the decision-making processes of software companies.
>
> **O3.** To support the operationalization of the specification and estimation of SSIs into a monitoring infrastructure that enables evidence-based decision making in software companies.

Each one of these objectives mapped directly to the problems P1-P3 addressed in this thesis. To reach these objectives, we followed a design science approach. As a result, we devised a method called SESSI (Specification and Estimation of Software Strategic Indicators). In addition, we defined a last objective O4 to explore the potential use of the resulting monitoring infrastructure and other related outputs from the SESSI method for advanced decision-making support.

> **O4.** To explore the use of the resulting assets from the SESSI method for enabling advanced evidence-based decision making.

This last objective helps us to get insights on the usefulness of the outputs of the SESSI method for forecasting the values of SSIs in order to provide advanced decision-making support.

These research objectives were translated to Research Questions (RQs) leading the development of this thesis. Table 1 shows the relation between the objectives and RQs.

Table 1 RQs and their connection with the objectives of the thesis

ID	RQ description	Objective
RQ1	*How are SSIs specified, estimated, and monitored for supporting evidence-based decision making in software companies?*	O1
RQ2	*How to support the specification, estimation, and monitoring of SSIs from exploiting corporate repositories and expert knowledge to promote evidence-based decision-making in software companies?*	O1, O2, O3
RQ3	*Is it feasible to apply the SESSI method to specify, estimate, and monitor SSIs for supporting evidence-based decision making in software companies?*	O1, O2, O3
RQ4	*Is it feasible to use the resulting assets from the SESSI method for enabling advanced decision-making support?*	O4

1.4 Methodological approach

This thesis has been carried out in the context of the Q-Rapids project[6], part of the Horizon 2020 program of the European Commission (Program H2020-EU.2.1.1. - Industrial Leadership - Leadership in enabling and industrial technologies - Information and Communication Technologies (ICT), Theme ICT-10-2016 - Software Technologies). The Q-Rapids project focused on providing improvements for the software development processes based on empirical information, data, and quality awareness, prioritizing the integration of Quality Requirements based on key SSIs. Therefore, the objectives of the thesis detailed in section 1.3 are partly framed in the context of the Q-Rapids project.

During the development of this thesis, the applicant participated in the Q-Rapids as a research member part of the GESSI research group from the Polytechnical University of Catalonia (UPC).

[6] https://www.q-rapids.eu

The research conducted in this thesis was based on the empirical-based approach devised in the Q-Rapids project, characterized by conducting empirical studies in the four software companies that conformed the Q-Rapids consortium. In particular, we based on the design-science approach to conceive the SESSI method. Further information on the methodological approach followed to reach the objectives can be found in Chapter 3.

1.5 Contributions of the thesis

The primary contribution of this thesis is a method for supporting the specification, estimation, and monitoring of SSIs, and thus support the evidence-based decision making in software companies. This method is named SESSI (Specification and Estimation of Software Strategic Indicators) and aims to tackle the problems stated in Section 1.2. The proposed SESSI method is provided with a set of specifically developed software artifacts, aimed to ease the application of the method, extracting, and exploiting the data from corporate repositories, and easing the estimation and monitoring of the SSIs.

Preliminary versions of the SESSI method were already published elsewhere (Manzano et al., 2018a, 2018b) and a consolidated version of the method was also published (Manzano et al., 2021). This consolidated version is summarized in Chapter 4, and detailed in Chapter 5, 6 and 7.

In addition, to demonstrate the usefulness of applying the SESSI method in a software company, we envisaged a way of forecasting the values of SSIs by greatly reusing the assets and infrastructure from the SESSI method. Results from a pilot case study in a software company provide positive insights that are detailed in Chapter 9.

1.6 Structure of the thesis

This thesis is structured in 10 chapters plus bibliography, annex, and appendices.

We provide an overview of each chapter of the thesis in Table 2.

Table 2 Overview of the chapters of the thesis.

Chapter	***Overview***
1	Introduces the thesis, provides the fundamentals including the context, problem, objectives, and contributions. A list of publications related to the thesis is also presented.
2	Details the state of the practice and state of the art of the specification, estimation, and monitoring of SSIs.
3	Describes the research context and the research methodology followed in this thesis, including a summary of the conducted research iterations for the design and evaluation of the SESSI method.
4	Provides an overview of the SESSI method, including its three main phases, i.e., the specification, estimation, and monitoring.
5	Describes the first phase of the SESSI method, corresponding to the specification of SSIs.
6	Describes the second phase of the SESSI method, corresponding to the estimation of SSIs.
7	Describes the third phase of the SESSI method, corresponding to monitoring SSIs.
8	Describes the case study performed in the context of the SESSI method summative validation.
9	Describes a way of forecasting the values of SSIs based on the resulting assets and infrastructure from the SESSI method in a software company.
10	Presents the conclusions and future work from the results obtained in this thesis.
11	Lists the references cited in this thesis.

1.7 List of publications

Table 3 summarizes the list of publications related to this thesis. The column "Related to" indicates the chapter of the thesis related to the publication.

Table 3 List of publications related to this thesis

Ref.	**Authors**	**Venue / Type**	**Title**	**Year**	**Related to**
(Manzano et al., 2018a)	Martí Manzano, Cristina Gómez, Claudia Ayala, Silverio Martínez Fernández, Prabhat Ram, Pilar Rodríguez, Marc Oriol	QuaRAP (Workshop)	Definition of the On-time Delivery Indicator in Rapid Software Development	2018	Chapter 3, 5
(Manzano et al., 2018b)	Martí Manzano, Emilia Mendes, Cristina Gómez, Claudia Ayala, Xavier Franch	PROMISE (Conference)	Using Bayesian Networks to estimate Strategic Indicators in the context of Rapid Software Development	2018	Chapter 3, 6
(Manzano et al., 2019)	Martí Manzano, Claudia Ayala, Cristina Gómez, Lidia López,	DSQA (Workshop)	A Software Service Supporting Software Quality Forecasting	2019	Chapter 9
(Manzano et al., 2021)	Martí Manzano, Claudia Ayala, Cristina Gómez, Antonin Abherve, Xavier Franch, Emilia Mendes	IST (Journal)	A Method to Estimate Software Strategic Indicators in Software Development: An Industrial Application	2021	Chapter 2, 3, 4, 5, 6, 7, 8
(López et al., 2021)	Lidia López, Martí Manzano, Cristina Gómez, Marc Oriol, Carles Farré, Xavier Franch, Silverio Martínez-Fernández, Anna Maria Vollmer	SciCO (Journal)	QaSD: A Quality-aware Strategic Dashboard for supporting decision makers in Agile Software Development	2021	Chapter 7, 8

(NA)	Martí Manzano, Claudia Ayala, Cristina Gómez, Antonin Abherve, Xavier Franch	Submitted to IST (Journal). Status: Major review. Revised manuscript resubmitted	An Industry-Academia Collaboration for Supporting Decision Making by Forecasting the Values of Software-Related Indicators from Corporate Repositories	(NA)	Chapter 9
(Pérez Torres et al., 2021)	Alberto Pérez, Cristina Gómez, Martí Manzano	UPC (Co-direction of a Final Degree Project)	Desenvolupament d'un sistema software per a la creació d'Indicadors Estratègics (SI) utilitzant Xarxes Bayesianes[7]	2021	Chapter 5, 6, 7
(Q-Rapids, 2019a)	Q-Rapids consortium	(Project deliverable)	Q-Rapids Deliverable D3.4	2019	Chapter 3
(Q-Rapids, 2019b)	Q-Rapids consortium	(Project deliverable)	Q-Rapids Deliverable D3.5	2019	Chapter 3

[7] "Development of a software system for the creation of Strategic Indicators (SI) using Bayesian networks."

2 State of the Practice and State of the Art

This chapter provides details on the state of the practice and state of the art related to the specification, estimation and monitoring of SSIs.

2.1 State of the Practice

As this thesis was performed in the context of the Q-rapids European project, to get insights about the state of the practice on the specification, estimation and monitoring of SSIs, a survey was performed at the beginning of the project in the context of the industrial partners of the Q-Rapids project.

It is important to remark that these partners provide an illustrative enough set of industrial profiles. Q-Rapids focused on four industrial partners with different profiles and sizes. These partners were also selected from different geographic regions in order to avoid cultural bias. Three of these partners are large companies within the European IT market, namely Nokia, Bittium and Softeam. The Small and Medium Enterprise (SME) sector is represented by the fourth partner, iTTi.

Details of the study performed to assess the state of the practice can be found in (Q-Rapids, 2018a) and was performed by the Polytechnical University of Catalonia (UPC)-GESSI team. Here a summary is provided to comprehend the state of the practice.

Goal: The goal of the survey was to gather information about the state of the practice on the specification, estimation, and monitoring of SSIs in the context of the four industrial partners of the Q-Rapids project.

Target: 4 industrial partners of the Q-Rapids project.

Instrument: Semi-structured interviews together with workshop sessions and in-situ observations were designed to gather data. The interview guide instrument can be consulted in Annex 1.

Procedure: All instruments were executed on each of the premises of the four industrial partners.

There were 12 interviews in total conducted by two or three researchers of the GESSI group. Each industrial partner provided 2 to 4 respondents for the interviews. Each interview lasted around one hour and were recorded and transcribed by an external company. In addition, 4 workshop sessions were held to get information about the state of the practice on specification, estimation and monitoring of SSIs.

Data analysis: Data analysis of the interview's responses and workshop sessions was performed through content analysis by researchers from the academic institutions of the Q-Rapids consortium.

Results regarding processes: The main results from the data analysis are:

- None of the industrial partners of the Q-Rapids project used any method or approach to assist them with the specification of SSIs to support their strategic decision-making processes. Instead, they stated that such indicators were implicit in the head of the decision makers.
- Decision makers confirmed that they did not explicitly specify such SSIs but took their decisions based on the information provided from their project management tools in use (e.g., Mantis, SonarQube, Redmine or Jenkins that provide operational indicators), and from their own expertise and intuition. A relevant problem that some of them emphasized regarding this approach was the dependency on the decision maker's experience. That is, if the person in charge of assessing the SSIs is not available, these can be incorrectly assessed or not assessed at all.
- Regarding the functionalities provided by some of their tools in use, they stated that such functionalities were quite limited. They missed:
    - That the indicators could be configured to fit the informational needs of the company or project.
    - A mechanism to aggregate heterogeneous information coming from several tools so they can get more strategic information.
- Decision makers remarked their interest on performing further assessment of relevant indicators, i.e., what-if analysis, diagnostic analysis, and scenario

assessments. This kind of analysis can be useful for decision makers to identify room for improvements, potential benefits, and risks.

Additionally, as respondents mentioned some tools in use or even some tools that they knew, the Q-Rapids team surveyed these tools to analyze their functionalities.

On the one hand, popular project management tools such as source code management platforms (e.g., Git, Subversion), source code analysis (e.g., Sonar, StyleCop) or backlog-focused tools (e.g., Redmine, Jira) provide predefined indicators to assess aspects of the software development lifecycle, (e.g., the sprint status -cumulative flow diagrams, burndown charts…- cumulative effort allocated, productivity metrics). However, the scope of these indicators was mainly considered as operational, as previously stated by the respondents of the interviews.

On the other hand, strategic decisions can be supported by Business Intelligence (BI) tools. These tools are commonly used in organizations for collecting, processing, and presenting information to decision makers in order to facilitate decision-making processes through business-related dashboards and reports (e.g., Tableau[8], Power BI[9], Qlik[10], and SAP BI[11]). However, the indicators and functionalities provided by these tools do not focus on the necessities of software companies (i.e., the data exploitation capabilities of BI tools do not precisely match with the usual tools used in the daily operation of software companies) nor the informational needs of a specific software company (i.e., their personalization features are limited) (Martínez-Fernández et al., 2018; Moe et al., 2012).

All in all, from the small-scale state of the practice survey, we conclude that the practical support for exploiting the surveyed software companies' data for specifying, estimating and monitoring SSIs for their particular contexts is needed.

[8] https://www.tableau.com

[9] https://powerbi.microsoft.com

[10] https://www.qlik.com/

[11] https://www.sap.com/products/technology-platform/bi-platform.html

2.2 State of the Art

Indicators have been traditionally proposed by academia and used by software companies to measure the success and quality aspects related to the fulfilment of their goals, their processes or products (Barone et al., 2011; IEEE, 2017). So, indicators have been recognized as essential assets for supporting decision making in software engineering contexts (IEEE, 2017). The concept of indicators is also referred with other terms, such as Key Performance Indicator (KPI), measurable aspects or factors (López et al., 2022).

In order to collect and assess evidence about the specification, estimation and monitoring of indicators in software engineering, and specifically in Agile and rapid Software Development (ASD), that was the predominant context of the Q-Rapids industrial partners; we surveyed the literature inspired on the procedures of systematic mapping studies (SMS). Mapping studies are a means of evaluating the state of research in a specific area (Budgen et al., 2008). We followed the guidelines for systematic literature reviews proposed by Kitchenham et al. (Kitchenham and Charters, 2007). However, in the searching process, instead of collecting studies from diverse databases, we used the set of papers previously collected by two SMS published recently as they covered the area of interest. The SMS used as baseline are:

- Baseline SMS 1: (López et al., 2022) that assessed 61 studies about metrics and quality-related indicators for ASD. This SMS was the natural baseline of this thesis as it was performed by the scientific team of the Q-Rapids project. It provides a comprehensive set of studies about quality indicators in ASD, that were the main interest of the industrial partners of the consortium.
- Baseline SMS 2: (Meidan et al., 2018) that assessed 462 studies related to software processes measurement. We decided to include this study even if it was not restricted to quality in ASD as the SMS 1 (López et al., 2022). The reason was to complement the SMS 1 (López et al., 2022) with a more generic work in order to make sure that we obtained a comprehensive enough set of papers.

Thus, the universe of papers initially considered was 462 + 61 papers = 523 papers.

To identify publications about specification, estimation and monitoring of indicators, the set of 523 papers from the two baseline SMSs were manually reviewed. In general, the author of this thesis, reviewed the titles and abstracts, and if necessary, skimmed the full text to decide if the paper was relevant for the purposes of each iteration, as described below. In case of doubts, the paper was discussed with the thesis' directors to decide the inclusion/exclusion of the paper.

Iteration 1: This iteration was dedicated to discard papers from the SMS 2 (Meidan et al., 2018) in order to target similar papers as the ones approached in the SMS 1 (López et al., 2022). Thus, the exclusion criteria in this iteration were:

EC1: Duplicated papers with respect to the SMS 1

EC2: papers published before 2001

EC3: papers not related to indicators

EC4: papers not related to ASD

As a result, 379 papers were discarded from the SMS 2 (Meidan et al., 2018).

Iteration 2: This iteration was aimed to review both SMS 1 and SMS 2 to discard papers that were not about tactical or strategic indicators. 25 papers were discarded from the SMS 2 (Meidan et al., 2018) while 15 papers were discarded from the SMS 1 (López et al., 2022).

Iteration 3: This iteration was aimed to review both SMS 1 and SMS 2 to discard papers that do not provide any insight about specification, estimation, and monitoring of indicators as these were the main topics of this thesis.

As a result of the selection process, we selected 22 studies from (López et al., 2022) and 58 from (Meidan et al., 2018). Hence, a total of 80 primary studies were selected to be further assessed. The selection procedure is summarized in Figure 3 and detailed in annex files (Manzano, 2022a, 2022b).

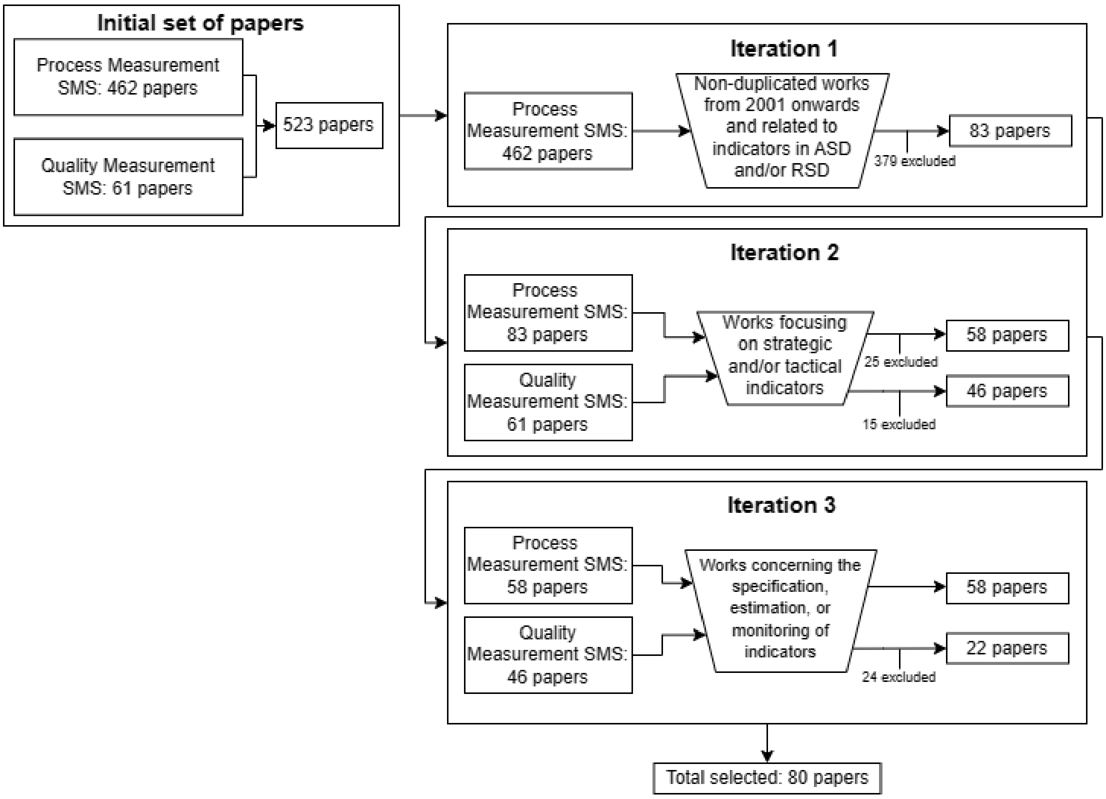

Initial set of papers
Process Measurement SMS: 462 papers
Quality Measurement SMS: 61 papers
523 papers
Iteration 1
Process Measurement SMS: 462 papers
Non-duplicated works from 2001 onwards and related to indicators in ASD and/or RSD
379 excluded
83 papers
Iteration 2
Process Measurement SMS: 83 papers
Quality Measurement SMS: 61 papers
Works focusing on strategic and/or tactical indicators
25 excluded
58 papers
15 excluded
46 papers
Iteration 3
Process Measurement SMS: 58 papers
Quality Measurement SMS: 46 papers
Works concerning the specification, estimation, or monitoring of indicators
58 papers
24 excluded
22 papers
Total selected: 80 papers

Figure 3 Filtering and selection process of primary studies

To analyse the papers, we developed our own criteria for gaining a broad understanding on how the literature deals with the specification, estimation, and monitoring of indicators.

Criteria used to assess the primary studies: The criteria were based on the state-of-the practice results and the experience from Q-Rapids researchers who are experts on indicators' specification and estimation. Each criterion was also supported by a question to ease the collection of the information from the papers. Table 4 summarizes the criteria used to assess the papers and categorize them according to the problems addressed by this thesis and introduced in Chapter 1 (P1, P2, P3).

Table 4 Criteria used to assess the selected primary studies in relation to the problems addressed in this thesis

Problems	**Criteria/ Question/ Expected value(s)**	**Rationale**
P1: Need of approaches that help software companies to define their own SSIs from the exploitation of their corporate data.	• Indicator: What is the name of the proposed indicator?	Provides the name given to the indicator.
	• Type of indicator: What is the type of the proposed indicator? (SSI or TACTICAL).	Defines the type of indicator as stated by the proposal.
	• Industrial evidence: Is the proposed indicator customized for a real company case? (YES/NO).	Provides insights about the real applicability of the proposal.
	• Data exploitation capabilities: Is the indicator explicitly defined based on available information/data? (YES/NO).	Provides insights on the appropriateness of the proposal to exploit corporate data.
	• Representation: How is the indicator specified? (QM-based, ad-hoc structure, formal language).	Provides insights on the mechanisms used to represent the indicator.
	• Reproducibility: Is there support available so that companies/organizations can render similar indicators for their own purposes? (YES/NO/PARTIAL).	Provides insights on the feasibility to define similar indicators in other contexts.
P2: Complex SSI estimation.	• Estimation procedure: The proposal provides a procedure for defining how to estimate the indicator? (YES/NO/PARTIAL).	The procedure to estimate an indicator (i.e., how to compute its values) is critical for its usage.
	• Expert-knowledge: The proposal considers expert knowledge? (YES/NO/PARTIAL).	Strategic decisions are greatly influenced by expert knowledge.
	• Missing Data: The proposal offers guidance on how to proceed with missing data? (YES/NO/PARTIAL).	Indicators assessment can be endangered by missing periods of data. So, it is important to know if the proposals deal with this.
	• Explainability. The proposal provides explainability facilities? (YES/NO/PARTIAL).	Strategic indicators use to aggregate relevant information and it is critical to provide a means to explain/justify its values.
P3: Need of approaches that support software companies to put forward the specification and estimation of their own SSIs for automatic monitoring purposes.	• Operationalization. The proposal offers details on how to operationalize the indicator for monitoring purposes? (YES/NO/PARTIAL).	(Semi)automatic monitoring requires the operationalization of the specification and assessment of indicators.

The following subsections detail the corresponding results.

2.3 Results

The results of the assessment of papers with respect to the criteria introduced in Table 4 are presented below and summarized in Table 5.

2.3.1 Indicators

We found a plethora of different indicators aimed to measure/estimate diverse aspects of the software lifecycle: from its development process to the customer satisfaction after being released or delivered. We found out that most of the works propose individual indicators aimed to tackle specific problems faced in the context of their application.

As the number of screened works is large (i.e., 80), as well as the total number of indicators presented in such works, we grouped the indicators using the same grouping criteria from (López et al., 2022). It is, we grouped the indicators presented in the assessed works into the following categories: schedule, risks, project success, productivity, product quality, process performance/quality, developer satisfaction, customer satisfaction, cost, and agility to assess their distribution. We have added the additional group "(generic)" for those works proposing frameworks, methodologies, or techniques instead of individual indicators. The results from this grouping are shown in Figure 4.

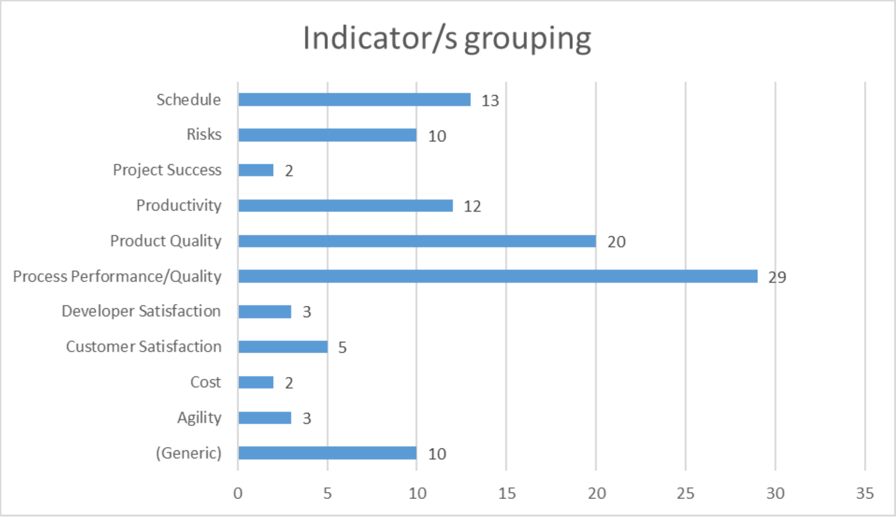


Figure 4 Groups of indicators discussed in the literature using the criteria from López et al. (López et al., 2022)

The most prevalent group is the “Process Performance/Quality”, with 29 appearances out of 109 indicators in total present in the 80 screened papers (e.g., (Chen et al., 2011; Dikici et al., 2012; Zhang et al., 2010). The second most prevalent group is “Product Quality”, with 20 appearances (e.g., (Syed-Mohamad and Md. Akhir, 2019; Vasilescu et al., 2015). The remaining groups of indicators refer to productivity (Bezerra et al., 2010; De Aquino Júnior and De Lemos Meira, 2009), or schedule (Bastarrica et al., 2017; Hearty et al., 2009). 10 out of the 80 assessed works contain generic proposals, in the sense that they do not focus on specific indicators but rather they provide instruments, methodologies, or methods to specify indicators according to the information needs (i.e., (Keser et al., 2013; Solingen et al., 2002)).

This result provides evidence on the interest of the community to tackle indicators related to quality.

2.3.2 Type of indicator

In order to understand the information level that the indicators covered, we recorded if they focus on strategic or tactical levels (operational indicators were discarded during the phase of papers’ filtering).

We found out that 39 out of 80 papers claimed to present strategic indicators. We observed that in most of the cases the strategic indicator was not the focus of the paper (except in 3 cases) but also other lower-level indicators related with the strategic indicator mentioned in the paper. Furthermore, we observed that some of the indicators that were referred as strategic in some works were also referred as tactical in other works. This led us to understand that the classification of the information level (tactical or strategic) of an indicator depends on the organization that uses them.

2.3.3 Industrial evidence

In order to analyze the industrial uptake of the assessed proposals, we recorded if the works provided industrial evidence. 46 out of 80 papers provided some kind of industrial evidence. Examples of works providing industrial evidence are Staron et al.’s proposal for the release readiness indicator (Staron et al., 2012), and the software project’s risk indicator by Chang (Chang, 2015). It is important to remark that despite presenting some kind of industrial evidence, most of the works do not describe the context in detail. This fact hinders the adoption of the proposals presented in such works, as interested companies may face difficulties to assess whether such proposals might fit their industrial context.

2.3.4 Data exploitation capabilities

To analyze whether the assessed works consider the data acquisition process for defining indicators, we screened details on how to get information/data from corporate repositories.

56 out of 80 works provided data exploitation capabilities while 24 papers did not provide enough details or do not provide any information at all. For example, Chen et al.'s (Chen et al., 2011) used data from configuration, test and requirement management systems for the specification and measuring of the software development process execution qualification rate.

Although a great deal of works provide some details on how to get the required data, it is important to remark that most of these works focus on very specific examples that do not face the challenges of the complexity of handling large amounts of heterogeneous data (Aranda and Easterbrook, 2005; Gren et al., 2017; Jørgensen and Sjøberg, 2004). Moreover, some authors remark the importance of not basing only on data, as this would discard the value expert knowledge and human factors can provide (Biddle et al., 2018; Matthies and Hesse, 2019; Sherdil and Madhavji, 1996).

2.3.5 Representation

To understand what type of artifacts or models are used to specify indicators, we registered such information.

53 out of 80 works use ad-hoc ways for specifying the indicators, motivated mostly by the specific information needs or restricted available data and information of the context in which they were specified. This is the case of works such as the continuous quality measurement by (Vassallo et al., 2018). The remaining 27 papers use instruments such as: Goal-Question-Metric (GQM) or variations (Shen and Ju, 2007; Solingen et al., 2002), ISO standards (Béland and Abran, 2012), Quality Models (QMs) (Staron et al., 2017) or formal languages (García et al., 2007).

These results show that there is not any kind of de facto standard representation for specifying indicators, but the use of quality models seems to be sound as they provide flexibility whilst keeping an agreed structure.

Figure 5 provides a summary of the results.

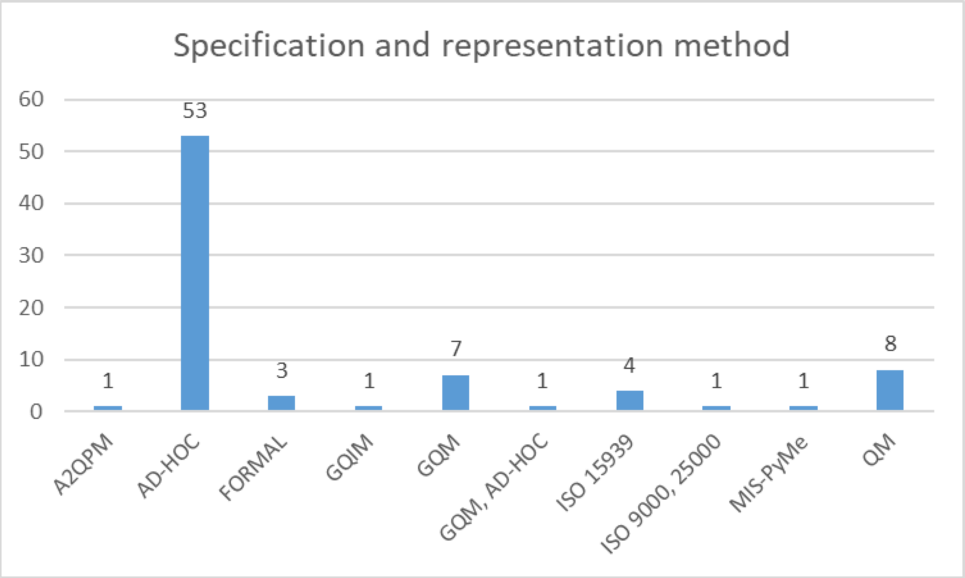


Figure 5 Classification of the assessed works according to the technique to specify and represent them

2.3.6 Reproducibility

To get evidence on the feasibility to reproduce the proposals in other contexts, we recorded if the assessed works provided explicit support for this.

Our results show that 41 out of 80 do not provide reproducibility details. An interesting insight we extracted is that authors usually consider the applicability of their proposed indicators for other contexts, however, they do not share further details but mere considerations of their applications. Examples of this type of works are the productivity indicator by W. Hao et al. (Hao et al., 2008) or the project velocity indicator for Extreme Programming (Hearty et al., 2009).

Only 15 papers provide reproducibility support while 24 provide only partial support. For example, the work providing an indicator for determining the success of software projects (Shashi et al., 2014), or the framework for the measurement of the software process execution qualification rate indicator (Chen et al., 2014).

In conclusion, although the indicators proposed in the assessed works may inspire other companies, their rationale cannot be realistically expected to be universal and reusable, because each company/organization may have its own intricacies yielding to specific needs and requirements (Carvallo et al., 2004). Hence, the general lack of support for reproducing and adapting the indicators proposed in these works can restrain their adoption in other contexts.

2.3.7 Estimation procedure

The procedure to estimate an indicator (i.e., compute or estimate the indicators' values) is critical for its usage, therefore, we registered if the assessed works provided such procedures.

The results show that 28 out of 80 works do not disclose the estimation's details while 30 works only do it partially (i.e., they mention the use of a technique or instrument for the estimation, but they do not provide further details).

An example of an assessed work providing details on the estimation procedure is the Barreto & Rocha's study (Barreto and Rocha, 2010), in which the authors provide a software projects' similarity indicator, while disclosing details on the characteristics, measures and calculations to measure such indicator. Other works (Hao et al., 2008; Zhang et al., 2010) provide the estimation procedure partially for the software productivity and trustworthiness indicators' estimation, respectively.

Similar to the insights extracted from the reproducibility criterion (section 2.3.6), the lack of details regarding the indicators' estimation procedure can restrain their adoption in other contexts.

2.3.8 Expert knowledge

Strategic decisions often require creativity and opportunistic inputs, and should be based on an accurate understanding of business processes and the products to release (Moe et al., 2012). These decisions are usually based on previous experiences, opinions and intuitions (Svensson et al., 2019), and the strategic decision-making process may extend over considerable periods of time (Moe et al., 2012). This is why providing capabilities to embed expert knowledge in the indicators' estimation procedures is important.

Despite the importance of expert knowledge, we found out that 45 out of 80 works do not consider the explicit use of expert knowledge, while 20 works do explicitly consider it. For instance, Kumar & Yadav (Kumar and Yadav, 2015) propose a model for estimating the risk of software projects using probabilistic models with domain expert's knowledge embedded. 15 works consider the use of expert knowledge partially (i.e., the work declares that expert knowledge may be used but do not provide further details). For instance, Zhang et al. (Zhang et al., 2014) provide a measurement model for software process risk measurement with partial expert knowledge consideration, as they do not explicitly consider it for the model construction, but rather only declare that statistical data or expertise may be used, without providing further details.

2.3.9 Missing data

Indicators' estimation can be endangered by missing periods of data. So, it is important to know if the proposals deal with this.

Our results show that only 8 out of 80 explicitly address this aspect. For instance, Freire et al. (Freire et al., 2018) propose a probabilistic model for the estimation of the process quality in Scrum-based projects. Their estimation model is able to deal with possible missing input data to estimate the indicator.

This result emphasizes the need of further work for dealing with missing data periods for estimating SSIs.

2.3.10 Explainability

Strategic indicators use to aggregate relevant information and it is critical to provide a means to explain/justify its values.

We found that only 12 out of 80 works provide an explicit means to ensure the indicators' estimation explainability. For instance, some works provide explainability basing the indicators' estimations on probabilistic graphical models (Perkusich et al., 2017; Zhang et al., 2014). Others works (Martinez-Fernandez et al., 2019, 2018) provide the estimations' explainability through web-based dashboards with drill-down capabilities.

Although few works addressed explainability, this factor is being demanded in order to add value to support decision systems (Svensson et al., 2019).

2.3.11 Operationalization

Monitoring indicators requires the operationalization of the specification and estimation of such indicators. To get evidence on how the primary studies deal with monitoring, we registered if the primary studies provide explicit support for enabling the monitoring of indicators.

We found that only 3 out of 80 works provided some kind of support by sharing some open-source software components. For instance, Martinez-Fernandez et al. (Martinez-Fernandez et al., 2019, 2018) provide free and open-source software components enabling the connection of several project management tools to their indicators' estimation procedures. These works are actually related to the Q-Rapids project. Other 6 works provide partial monitoring support. It was considered partial because they mentioned that indicator's monitoring was done but they did not provide further information about it. For instance, Staron et al. (Staron et al., 2013, 2012) explicitly report monitoring support for their proposed indicators (software stability and release readiness, correspondingly). However, the authors do not share their developed software artifacts to enable the indicators' monitoring as they are developed in the context of specific companies.

As a result, although the software components and instructions that have been shared are valuable, the available support is considered scarce as these artifacts do not allow a company to put forward a comprehensive monitoring infrastructure.

Table 5 Characterization of the assessed works

	P1							P2				P3
REF.	Indicator/s	Type	Industr. Eviden.	Data exploit.	Repres.	Reprod.	Estim. Proc.	Expert Know.	Missing Data	Explain.	Operat.
(Solingen et al., 2002)	(Method for eliciting and defining indicators and metrics)	STRATEGIC, TACTICAL	NO	YES	GQM	YES	NO	YES	NO	NO	NO
(Schackmann et al., 2009)	Software process QM	STRATEGIC, TACTICAL	NO	YES	QM	PARTIAL	PARTIAL	PARTIAL	NO	ND	NO
(Gucegliogu and Demirors, 2005)	Process Quality	TACTICAL	YES	NO	QM	PARTIAL	PARTIAL	YES	NO	YES	NO
(Tüysüz and Kahraman, 2006)	Project Risk	STRATEGIC	YES	NO	AD-HOC	NO	YES	YES	NO	NO	NO
(Shen and Ju, 2007)	Agile aspects (ROI, Productivity, Quality, Adaptability, Innovation)	STRATEGIC, TACTICAL	YES	NO	GQM	NO	YES	PARTIAL	NO	NO	NO
(García et al., 2007)	Process measures	TACTICAL	NO	NO	FORMAL	YES	NO	YES	NO	YES	NO
(García et al., 2003)	Process measures	TACTICAL	NO	NO	FORMAL	YES	NO	YES	NO	YES	NO
(García et al., 2006)	Process measures	TACTICAL	NO	NO	FORMAL	YES	NO	YES	NO	YES	NO
(Tang, 2008)	Health for ASD	STRATEGIC, TACTICAL	NO	NO	AD-HOC	PARTIAL	YES	PARTIAL	NO	NO	NO
(Mahnic and Zabkar, 2008)	Progress metrics	TACTICAL	YES	YES	AD-HOC	PARTIAL	PARTIAL	NO	NO	NO	NO
(Kojima et al., 2008)	Risk in early development stages	STRATEGIC, TACTICAL	NO	YES	AD-HOC	NO	PARTIAL	NO	NO	NO	NO
(Hao et al., 2008)	Productivity	STRATEGIC, TACTICAL	NO	YES	AD-HOC	NO	PARTIAL	NO	NO	NO	NO
(Quah and Liew, 2008)	Software Readiness	STRATEGIC, TACTICAL	NO	YES	AD-HOC	NO	PARTIAL	NO	NO	NO	NO
(Lin and Huang, 2009)	Framework (metrics for software process)	TACTICAL	NO	YES	GQM	NO	PARTIAL	PARTIAL	NO	NO	NO
(Bezerra et al., 2010)	General Project Productivity	TACTICAL	YES	YES	AD-HOC	NO	YES	NO	NO	NO	NO
(Hearty et al., 2009)	Project Velocity in Extreme Programming	STRATEGIC, TACTICAL	YES	YES	AD-HOC	NO	PARTIAL	PARTIAL	YES	YES	NO
(De Aquino Júnior and De Lemos Meira, 2009)	Productivity	TACTICAL	YES	NO	AD-HOC	YES	PARTIAL	PARTIAL	NO	NO	NO
(Zhang et al., 2010)	Trustworthiness of Software Processes	TACTICAL	NO	YES	AD-HOC	PARTIAL	PARTIAL	PARTIAL	NO	NO	NO
(Shawky and Ali, 2010)	Agility of SD Processes	TACTICAL	NO	YES	AD-HOC	NO	NO	NO	NO	NO	NO
(Wu et al., 2010)	Software Reliability	TACTICAL	NO	YES	AD-HOC	PARTIAL	PARTIAL	PARTIAL	NO	NO	NO
(Barreto and Rocha, 2010)	Software Projects Similarity	STRATEGIC, TACTICAL	NO	YES	AD-HOC	PARTIAL	YES	PARTIAL	NO	NO	NO
(Petersen and Wohlin, 2011)	Flow in Lean Software Dvelopment	TACTICAL	YES	YES	AD-HOC	NO	YES	NO	NO	NO	NO
(Dikici et al., 2012)	Process Quality	STRATEGIC, TACTICAL	YES	ND	QM	NO	NO	PARTIAL	NO	YES	NO
(Castro et al., 2012)	Software Value indicators	STRATEGIC	NO	NO	AD-HOC	NO	YES	NO	NO	NO	NO
(Lami et al., 2013)	Methodology for sustainability indicators	STRATEGIC, TACTICAL	NO	NO	GQM	YES	NO	ND	NO	NO	NO
(Ikemoto et al., 2013)	Reliability	TACTICAL	NO	YES	AD-HOC	NO	PARTIAL	NO	NO	NO	NO
(Shashi et al., 2014)	Success of Software Projects	STRATEGIC	NO	NO	AD-HOC	YES	YES	NO	NO	NO	NO
(Zhang et al., 2014)	Risk of the process	STRATEGIC, TACTICAL	NO	YES	AD-HOC	NO	PARTIAL	PARTIAL	YES	YES	NO
(Tarhan and Yilmaz, 2014)	Product Quality, Performance, Process Performance, System Test Phase Performance	STRATEGIC, TACTICAL	YES	YES	GQM	NO	NO	NO	NO	NO	NO
(Shahnewaz and Ruhe, 2014)	Release Readiness	STRATEGIC, TACTICAL	NO	YES	GQM	PARTIAL	PARTIAL	YES	NO	NO	NO
(Kumar and Yadav, 2015)	Risk for software projects	STRATEGIC, TACTICAL	NO	NO	AD-HOC	PARTIAL	YES	YES	YES	YES	NO
(Chang, 2015)	Software Risk	STRATEGIC, TACTICAL	YES	YES	AD-HOC	PARTIAL	PARTIAL	NO	NO	NO	NO
(Padmini et al., 2015)	Product Quality, Team Productivity, Predictability	TACTICAL	NO	ND	AD-HOC	NO	NO	ND	NO	NO	NO
(Yang et al., 2009)	Process Trustworthiness	TACTICAL	NO	ND	AD-HOC	NO	NO	ND	NO	NO	NO
(Khokhar et al., 2010)	Guidelines for software processess monitoring	TACTICAL	NO	ND	AD-HOC	PARTIAL	NO	ND	NO	NO	NO
(Monteiro and De Oliveira, 2011)	Indicators for process performance analysis	TACTICAL	YES	YES	AD-HOC	NO	NO	NO	NO	NO	NO
(Chen et al., 2014)	Process execution qualification rate	STRATEGIC, TACTICAL	YES	YES	AD-HOC	PARTIAL	PARTIAL	NO	NO	NO	NO
(Sunetnanta and Choetkiertikul, 2012)	Process capability maturity and risk	STRATEGIC, TACTICAL	YES	YES	AD-HOC	NO	YES	NO	NO	NO	NO
(Chen et al., 2014)	Process execution qualification rate	TACTICAL	NO	YES	AD-HOC	YES	YES	PARTIAL	NO	NO	NO
(List et al., 2005)	Customer Satisfaction metrics	TACTICAL	YES	YES	AD-HOC	NO	NO	YES	NO	NO	NO
(Johnson et al., 2005)	Development telemetry	TACTICAL	YES	YES	AD-HOC	YES	YES	NO	NO	NO	YES

(Staron and Meding, 2009)	(Method)	STRATEGIC, TACTICAL	YES	YES	ISO 15939	PARTIAL	NO	ND	NO	NO	NO
(Staron et al., 2011)	(Method)	STRATEGIC, TACTICAL	YES	YES	ISO 15939	PARTIAL	NO	ND	NO	NO	NO
(Díaz-Ley et al., 2008)	(Method for eliciting and defining indicators and metrics)	STRATEGIC, TACTICAL	YES	YES	MIS-PyMe	YES	NO	YES	NO	NO	NO
(Béland and Abran, 2012)	(Framework)	STRATEGIC, TACTICAL	YES	YES	ISO 15939	PARTIAL	NO	ND	NO	NO	NO
(Keser et al., 2013)	(Measurement Tool)	STRATEGIC, TACTICAL	YES	YES	GQIM	PARTIAL	NO	ND	NO	ND	NO
(Wagner and Dürr, 2006)	Method for value-based planning and monitoring of systems engineering projects	STRATEGIC, TACTICAL	NO	YES	AD-HOC	PARTIAL	NO	ND	NO	NO	NO
(Wahyudin and Tjoa, 2007)	Developers' events	TACTICAL	NO	YES	AD-HOC	NO	NO	NO	NO	NO	NO
(Mahnič and Vrana, 2007)	Performance metrics of the software development process	TACTICAL	YES	YES	AD-HOC	NO	YES	NO	NO	NO	NO
(Colombo et al., 2008)	Measures and KPIs of software development processes	TACTICAL	YES	YES	GQM	NO	PARTIAL	YES	NO	NO	PARTIAL
(Wu et al., 2009)	Metrics for Software Project Management	TACTICAL	NO	YES	AD-HOC	NO	NO	ND	NO	NO	NO
(Cuadrado-García et al., 2011)	Project Evolution	TACTICAL	NO	YES	AD-HOC	NO	PARTIAL	NO	NO	NO	NO
(Staron and Meding, 2011)	Bottlenecks in ASD and Lean software development projects	TACTICAL	YES	YES	AD-HOC	PARTIAL	PARTIAL	YES	NO	NO	NO
(Tarhan and Demirors, 2012)	(Method for eliciting and defining indicators and metrics)	STRATEGIC, TACTICAL	YES	YES	A2QPM	PARTIAL	NO	YES	NO	NO	NO
(Staron et al., 2017)	Quality Model for KPIs	STRATEGIC, TACTICAL	YES	NO	QM	NO	NO	YES	NO	ND	NO
(Matthies et al., 2016)	Agile violations and conformance metrics	TACTICAL	NO	YES	AD-HOC	YES	YES	NO	NO	NO	PARTIAL
(Staron et al., 2018)	Software development process metrics and indicators	STRATEGIC, TACTICAL	YES	YES	AD-HOC	NO	PARTIAL	ND	YES	NO	PARTIAL
(Perkusich et al., 2017)	Process Quality	TACTICAL	YES	NO	AD-HOC	YES	YES	YES	YES	YES	NO
(Perkusich et al., 2015)	Project deviation	TACTICAL	YES	YES	AD-HOC	NO	NO	NO	NO	NO	NO
(Salo et al., 2002)	Approach for supporting software development metrics	STRATEGIC, TACTICAL	NO	NO	GQM, AD-HOC	PARTIAL	NO	ND	NO	NO	NO
(Ilieva et al., 2004)	Project Cost Change, Relative Cost Deviation, Customer satisfaction, Developer satisfaction, Defect fixing effort, Feature throughput, Defect rate, Relative Schedule Deviation	STRATEGIC, TACTICAL	YES	ND	AD-HOC	NO	PARTIAL	YES	NO	NO	NO
(Layman et al., 2004)	Customer satisfaction, Team morale, Programmer Productivity, Pre-release quality, Post-release quality	STRATEGIC, TACTICAL	YES	ND	AD-HOC	NO	PARTIAL	ND	NO	NO	NO
(Mann and Maurer, 2005)	Customer satisfaction, overtime	TACTICAL	YES	NO	AD-HOC	NO	YES	NO	NO	NO	NO
(Martinez-Fernandez et al., 2018)	Issues' velocity, Code Quality, Blocking code, Testing Status, Software Stability (bugs), Product Quality, Time-to-Market	STRATEGIC, TACTICAL	YES	YES	QM	YES	YES	PARTIAL	YES	YES	YES
(Martinez-Fernandez et al., 2019)	Issues' velocity, Code Quality, Blocking code, Testing Status, Software Stability (bugs), Product Quality, Time-to-Market	STRATEGIC, TACTICAL	YES	YES	QM	YES	YES	PARTIAL	YES	YES	YES
(Ericsson and Sweden, 2017)	Development Progress metrics	TACTICAL	YES	YES	AD-HOC	NO	PARTIAL	YES	NO	NO	NO
(Staron et al., 2012)	Release Readiness, Time to Release	STRATEGIC, TACTICAL	YES	YES	ISO 15939	PARTIAL	YES	NO	NO	NO	NO
(Staron et al., 2014)	Development progress, Trend development, Architecture stability, Product Reliability, Internal Quality, External Quality, Release Readiness, Development Progress	STRATEGIC, TACTICAL	YES	YES	ISO 9000, 25000	NO	PARTIAL	NO	NO	PARTIAL	PARTIAL
(Vasilescu et al., 2015)	Team Productivity, Code Quality	TACTICAL	NO	YES	AD-HOC	NO	PARTIAL	NO	NO	NO	NO
(Antinyan et al., 2014)	Relative Risk	TACTICAL	YES	YES	AD-HOC	YES	YES	NO	NO	NO	NO
(Bakota et al., 2012)	Development Cost	TACTICAL	YES	YES	AD-HOC	NO	PARTIAL	NO	NO	NO	NO
(Boldt et al., 2017)	Corrected risks, Postponed risks, Risk value, Unhandled risks	TACTICAL	YES	NO	AD-HOC	NO	NO	YES	NO	NO	NO
(Çalıklı Chalmers and Meding, 2018)	Release Quality	TACTICAL	YES	YES	AD-HOC	NO	NO	PARTIAL	NO	NO	NO
(Olague et al., 2006)	Software design stability	TACTICAL	YES	YES	AD-HOC	NO	YES	NO	NO	NO	NO
(Roden et al., 2007)	Software design stability	TACTICAL	YES	NO	AD-HOC	NO	NO	NO	NO	NO	NO
(Staron et al., 2013)	Architecture stability	TACTICAL	YES	YES	AD-HOC	PARTIAL	PARTIAL	NO	PARTIAL	NO	PARTIAL
(Syed-Mohamad and Md. Akhir, 2019)	Pull Request Release Readiness	TACTICAL	NO	YES	AD-HOC	PARTIAL	PARTIAL	NO	PARTIAL	NO	PARTIAL
(Vassallo et al., 2018)	Continous Code Quality	TACTICAL	NO	YES	AD-HOC	NO	PARTIAL	NO	NO	NO	NO
(Manzano et al., 2018b)	Software Stability, Product Quality, Code Quality, Time-To-Market, Estimated Effort, Customer Satisfaction	STRATEGIC, TACTICAL	YES	YES	QM	PARTIAL	YES	YES	YES	YES	NO
(Manzano et al., 2018a)	On-Time Delivery	STRATEGIC, TACTICAL	YES	YES	QM	NO	PARTIAL	YES	NO	NO	NO

2.4 Conclusions

After analyzing the state of the practice and state of the art on specification, estimation and monitoring of SSIs, and to answer **RQ1,** we can conclude that although there exist some published proposals that deal with some of the aspects raised by the practitioners, none of these proposal fully addresses the whole list of main problems.

The criteria most commonly left behind are: monitoring support (3% of the works consider it), the addressing of missing data (10% of the works), explainability capabilities (15% of the works), reproducibility (18% of the works), and consideration of expert knowledge (25% of the works).

We can conclude there exists a lack of practical support for addressing the specification, estimation, and monitoring of indicators related to strategic decisions for the particular contexts of software companies.

3 Research Methodology

This chapter describes the context of this work and the research methodology devised to reach the objectives of this thesis.

3.1 Research Context

3.1.1 The Q-Rapids project

Most of the contributions of this thesis were devised in the context of the Q-Rapids project (Q-Rapids, 2019c). The Q-Rapids project is a European-funded project aimed to improve software quality and software development processes in general, through an empirical-based, data-driven, and quality-aware rapid software development methodology. Such methodology prioritizes the integration of appropriate Quality Requirements in software life-cycle, basing such integration on key indicators presented to decision makers through a strategic dashboard.

The Q-Rapids project had a duration of 36 months and ran from November 2016 to October 2019. It was composed of a multidisciplinary consortium of seven organizations from five countries, including academic and industrial organizations. Regarding the academic partners, there were three institutions: the Polytechnical University of Catalonia (UPC), the University of Oulu and the Fraunhofer Institute for Experimental Software Engineering. As industrial partners, the project had four software companies with different profiles and sizes: Nokia[12], Bittium[13], Softeam[14] and iTTi[15].

The following paragraphs provide a summarized description of each industrial partner.

- Bittium is a company specialized in the development of reliable, secure communications and connectivity solutions, leveraging its 30-year legacy of expertise in advanced radio communication technologies. Bittium provides innovative products and customized solutions based on its product platforms and R&D services. Complementing its communications and connectivity solutions, Bittium offers proven information security solutions for mobile devices and portable computers. Bittium offers its customers also healthcare technology products and services in biosignal measuring in the areas of cardiology, neurology, rehabilitation, occupational health and sports medicine.
- iTTi belongs to the Small and Medium Enterprise (SME) sector working in IT and telecommunications fields. The activities of iTTi can be grouped into three categories: technical consulting in the area of telecommunications and IT, applied R&D in the area of IT and telecommunications and development of innovative applications and software solutions (e.g., in crisis management, health and space sectors). iTTi carried out research activities in the following programmes: EU-funded initiatives (Horizon 2020, FP7, FP6 and FP5), European Defence Agency (EDA) programmes as well as Action Grant CIPS II and NATO Industrial Advisory Group studies. iTTi has been also involved in the European Space Agency (ESA) projects. In R&D activities, the company cooperates closely with numerous universities and research institutes based in Poland as well as around Europe.
- Nokia is a large company committed to the innovation in the field of telecommunications. Powered by the research and innovation of Nokia Bell Labs, it serves communications service providers, governments, large enterprises and consumers, with end-to-end portfolio of products, services and

[12] https://www.nokia.com

[13] https://www.bittium.com

[14] https://www.softeamgroup.fr

[15] https://www.itti.com.pl

licensing. From the enabling infrastructure for 5G and the Internet of Things, to emerging applications in virtual reality and digital health, Nokia is shaping the future of technology to transform the human experience.

- Softeam is a large software company dedicated to providing business consulting through services and solutions in strategy, consulting, finance, digital, big data, artificial intelligence, analytics, performance, and operations. Their product line is composed of several tools and extensions based on a large set of technologies from heavy client to cloud web-based application or specialized server. Among these, Softeam develops and maintains Modelio, the last generation of a 25-year-old product line of a model-driven tool suite dedicated to expressing and managing requirements, modelling software architectures, building accurate UML models, generating a full range of documentation and automating application code production for several languages.

These partners provided use cases from heterogeneous domains: networks and telecommunications, defense and military systems, technical consulting, and software solutions. Their respective use-cases were considered business-critical for each industrial partner, and they participated in the Q-Rapids consortium in order to improve their current situations.

An overview of the general objectives and further decomposed scientific objectives of the Q-Rapids project is shown in Figure 6.

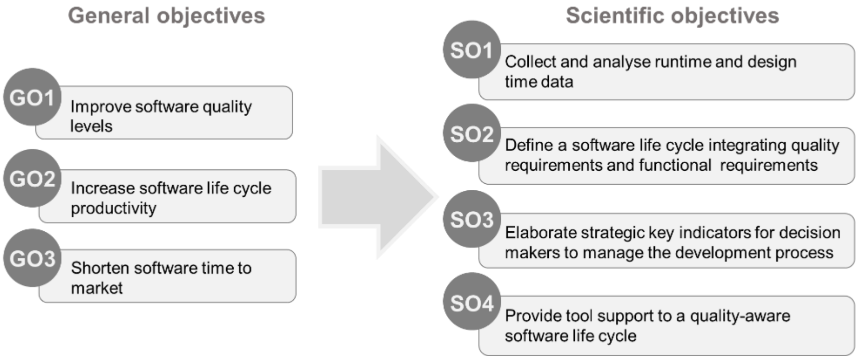

General objectives
GO1
Improve software quality levels
GO2
Increase software life cycle productivity
GO3
Shorten software time to market
Scientific objectives
SO1
Collect and analyse runtime and design time data
SO2
Define a software life cycle integrating quality requirements and functional requirements
SO3
Elaborate strategic key indicators for decision makers to manage the development process
SO4
Provide tool support to a quality-aware software life cycle

Figure 6 General and scientific objectives of the Q-Rapids project

Among the scientific objectives shown in Figure 6, **SO3** contemplates the elaboration of SSIs that serve as evidence to improve the development process, including the product/s under development. The main objectives of this thesis relate directly with this scientific objective of the Q-Rapids project.

3.2 Research Methodology

The research methodology conceived to reach the main objectives of the thesis (as described in Chapter 1) was influenced by the industry-academia collaboration context from the Q-Rapids project. The Q-Rapids industrial partners provided heterogeneous use cases that were used to study the problems and articulate, refine, and validate the proposed solutions of this thesis.

As stated by Wohlin and Runeson (Wohlin and Runeson, 2021), the selection of appropriate research methodologies for dealing with industry-academia collaborations is essential to maximize their benefits and ensure their success. In particular, there are three candidate research methodologies for building and evaluating solutions developed to address an industrial challenge, namely: action research (Avison et al., 1999; Elden and Chisholm, 1993), design-science (Wieringa, 2014) and technology transfer models (Mikkonen et al., 2018). They are usually characterized by their aim to make a change of practice (action research), or produce an artifact (design science), or transfer knowledge about some practice (technology transfer models). Although there is some debate about their similarities and differences, Wohlin and Runeson (Wohlin and Runeson, 2021) suggest that their selection should be based on the primary objective and scope of the collaborations. They also emphasize that these three methodologies are complementary, so that elements from other research methodologies may influence the implementation of the chosen research methodology.

In this thesis, we adopted the design science research methodology as the high-level frame to articulate a method (i.e., an artifact) that help to ameliorate the problems introduced in Chapter 1. The method was conceived, refined, and validated in the contexts of the Q-Rapids' industrial partners. Thus, in line with the design science cycle described by Wieringa (Wieringa, 2014), we envisaged three stages for conceiving the method: 1) Problem Investigation. 2) Solution Design. 3) Solution Validation.

In addition, given that the effort required to put forward the SESSI method in a software company resulted considerable, we contemplated the provision of insights on the potential benefits that the resulting assets from the SESSI method can bring to the software companies to promote evidence-based decision making. Hence, we promoted Stage 4. In this stage, we explored the feasibility of using the resulting models, data and infrastructure from the SESSI method for forecasting the values of SSIs.

Figure 7 shows an overview of the high-level research design of this thesis. Stages 1-3 corresponds to the suggested stages from design-science (Wieringa, 2014) for articulating the SESSI method while Stage 4 was aimed to provide insights on the usefulness of the resulting assets from the SESSI method for forecasting purposes.

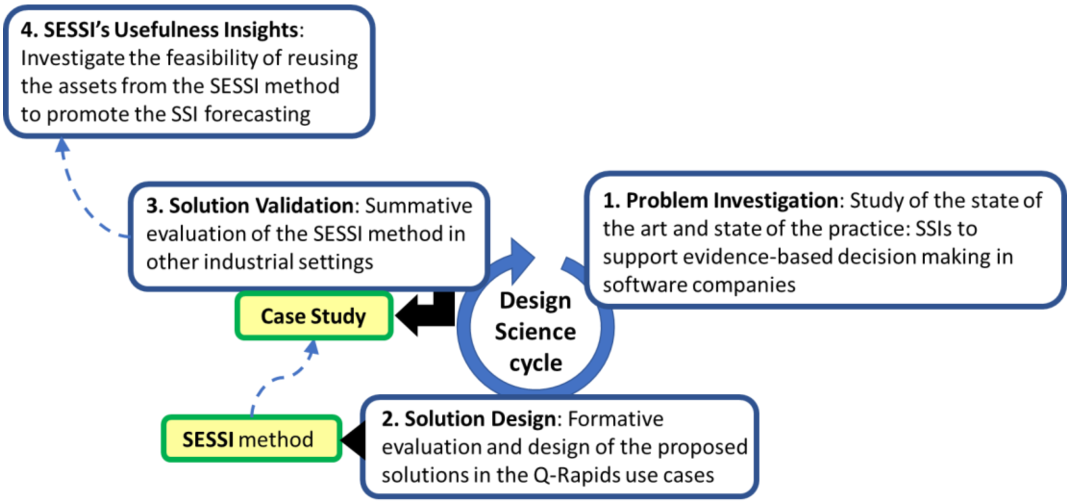

4. SESSI's Usefulness Insights: Investigate the feasibility of reusing the assets from the SESSI method to promote the SSI forecasting
3. Solution Validation: Summative evaluation of the SESSI method in other industrial settings
1. Problem Investigation: Study of the state of the art and state of the practice: SSIs to support evidence-based decision making in software companies
Case Study
Design Science cycle
SESSI method
2. Solution Design: Formative evaluation and design of the proposed solutions in the Q-Rapids use cases

Figure 7 High-level research design of this thesis

The following subsections provide details of each stage.

3.2.1 Stage 1-Problem Investigation

This stage was led by **RQ1** *"How are SSIs specified, estimated, and monitored for supporting evidence-based decision making in software companies?"*. The focus was on indicators as they are the most commonly used artifacts in software engineering to represent relevant information able to inform decisions (apart from the tacit expert knowledge) (IEEE, 2017). So, we conducted a study to investigate the state of the practice in the context of the Q-Rapids industrial partners regarding the specification and estimation of SSIs.

To investigate the state of the practice and the state of the art regarding the specification, estimation, and monitoring of SSIs for supporting the evidence-based decision making we proceeded as follows.

a) To investigate the state of the art, we performed a literature study. We found out that most of the screened works proposed ad-hoc defined indicators (not necessarily SSIs) for specific contexts, without providing enough detail to ease their generalization to other contexts nor guidance to specify, estimate, and monitor any SSIs according to any company's needs. Furthermore, although the existing works address individual or subset of the problems presented in Chapter 1, they do not address them altogether.

b) To analyse the state of the practice, we performed a survey in the context of the four industrial partners of the Q-Rapids project, in-situ observations and workshop sessions. They were aimed to explore the use of SSIs to deal with the decision-making processes in these companies. The main extracted insight was that these companies did not use specific approaches for the specification, estimation, and monitoring of SSIs but relied on their own knowledge, intuition, and experience and also used some functionalities provided by their project management tools. In addition, the participants remarked that the indicators and functionalities provided by their existing tools were quite limited and further support was needed to improve their strategic and tactical decision-making processes).

The results from conducting this stage to answer **RQ1** are presented in Chapter 2.

3.2.2 Stage 2-Solution Design

This stage focused on devising a solution to ameliorate the main problems found in Stage 1 related to the specification, estimation, and monitoring of SSIs for supporting evidence-based decision making in software companies. Stage 2 was led by **RQ2** *“How to support the specification, estimation, and monitoring of SSIs from exploiting corporate repositories and expert knowledge to promote evidence-based decision-making in software companies?”* For this, we followed an action-research approach (Avison, 2003; Avison et al., 1999) in the context of the four Q-Rapids industrial partners to formulate and apply solution attempts to the main detected problems from Stage 1. From the lessons learnt and insights gained from such formative processes, we finally articulated an integral solution, i.e., the SESSI method. Preliminary results from the formative stage of the SESSI method were published in (Manzano et al., 2018a, 2018b) and are briefly described in the following subsections. The action-research iterations conducted to formulate the SESSI method are summarized in Figure 8 and described below.

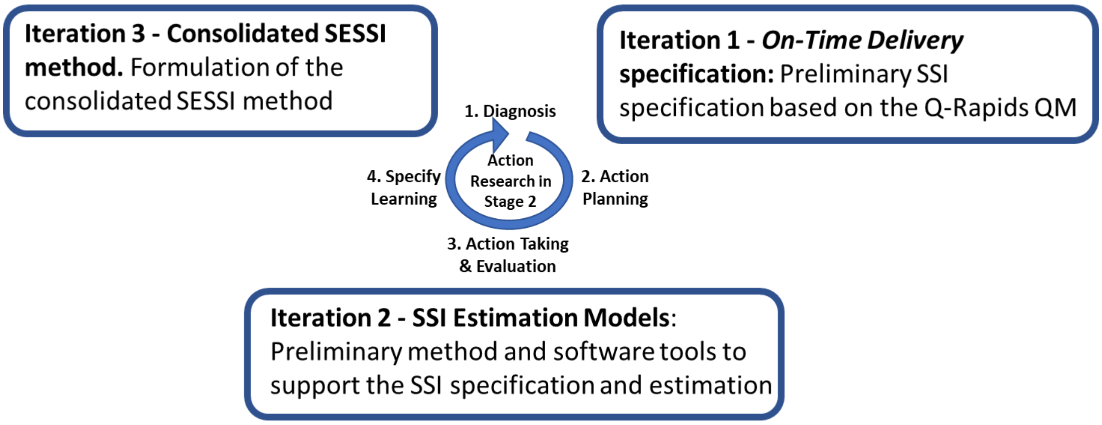

Iteration 3 - Consolidated SESSI method. Formulation of the consolidated SESSI method
Iteration 1 - *On-Time Delivery* specification: Preliminary SSI specification based on the Q-Rapids QM
1. Diagnosis
2. Action Planning
3. Action Taking & Evaluation
4. Specify Learning
Action Research in Stage 2
Iteration 2 - SSI Estimation Models: Preliminary method and software tools to support the SSI specification and estimation

Figure 8 Action-Research iterations conducted as part of Stage 2

3.2.2.1 Stage 2-Iteration 1: *On-Time Delivery* specification

This iteration focused on providing a generic specification of an SSI that was relevant for the Q-Rapids industrial partners: *On-Time Delivery* SSI.

This SSI was chosen and agreed upon by the Q-Rapids industrial partners. Its specification was based on literature reviews and eliciting the industrial partners' needs through workshops and interviews specifically designed for this purpose. As a result, we provided a generic specification of the SSI that can be adapted by interested software companies according to their individual needs.

The *On-Time Delivery* SSI aims to characterize and ease the detection of development problems, in order to prevent delivery delays, and to estimate the additional time needed when software requirements (especially quality requirements) are considered. We specified the *On-Time Delivery* SSI, inspired on the structure proposed in Q-Rapids Quality Model (QM) (Martinez-Fernandez et al., 2018) that suggests a hierarchical structure (including SSI, factors and metrics). Such structure was convenient for ensuring the interpretability and explainability of the SSI.

The *On-Time Delivery* was specified as an SSI decomposed into 5 factors, each of them further decomposed in a set of corresponding metrics. These metrics, at their turn were specified as normalized values using utility functions in the [0,1] interval, computed from raw data from corporate repositories such as GitHub and Redmine. The estimation method to compute the factors and the SSI itself would be based on an aggregation formula using weighted sums (w_i and w_{ij} weights).

Figure 9 shows a graphical summary of the *On-Time Delivery* SSI specification.

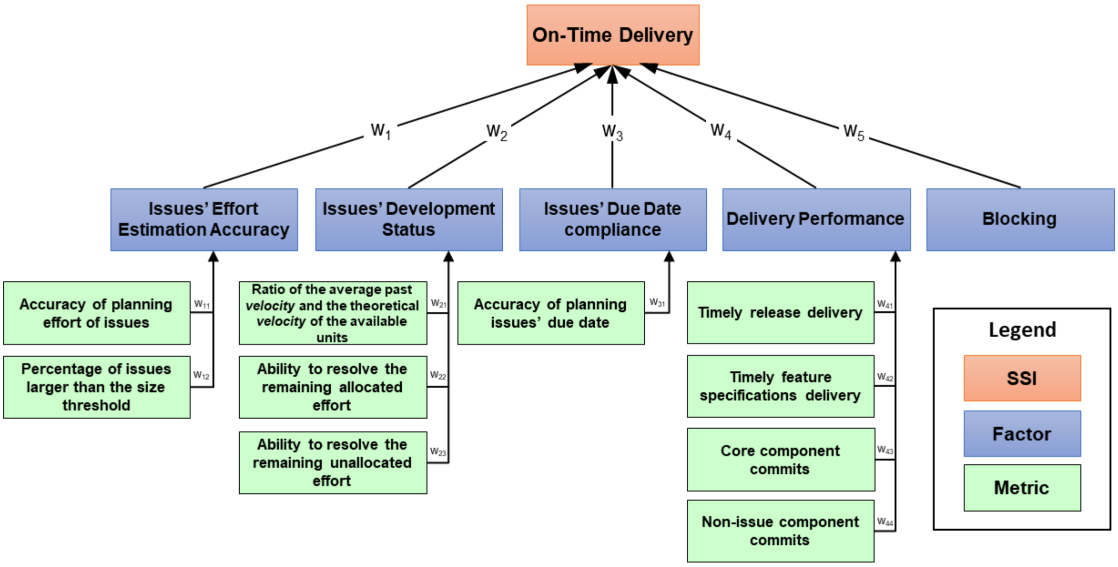

On-Time Delivery
W1
W2
W3
W4
W5
Issues' Effort Estimation Accuracy
Issues' Development Status
Issues' Due Date compliance
Delivery Performance
Blocking
Accuracy of planning effort of issues
w11
Percentage of issues larger than the size threshold
w12
Ratio of the average past velocity and the theoretical velocity of the available units
w21
Ability to resolve the remaining allocated effort
w22
Ability to resolve the remaining unallocated effort
w23
Accuracy of planning issues' due date
w31
Timely release delivery
w41
Timely feature specifications delivery
w42
Core component commits
w43
Non-issue component commits
w44
Legend
SSI
Factor
Metric

Figure 9 Summary of the *On-Time Delivery* specification resulting from the first research iteration

More details on the *On-Time Delivery* SSI specification, and on this first iteration in general were published and can be consulted in (Manzano et al., 2018a).

3.2.2.2 Stage 2-Iteration 2: SSI estimation models

This second iteration focused on SSIs estimation. It was aimed to devise a way to estimate the SSIs previously specified.

To do so, we designed a preliminary method and software supporting tools to gather and combine corporate data and knowledge to enable the estimation of meaningful SSIs that would help to inform relevant decisions in software companies. The method supports the specification and estimation of SSIs using probabilistic models. The method was inspired on the Bayesian Networks (BNs) approach EKEBN (Expert-based Knowledge Engineering of Bayesian Networks) (Mendes, 2014; Mendes et al., 2018). As a result, SSI are specified, and estimation models based on BNs are constructed for enabling SSI estimation and monitoring.

The formative evaluation of the method and its supporting tools included the elaboration of two different SSIs in the context of the Q-Rapids industrial partners. First, we focused on the *On-Time Delivery* SSI defined in the first research iteration. An excerpt of the resulting BN probabilistic graphical estimation model for the *On-Time Delivery* SSI is shown in Figure 10.

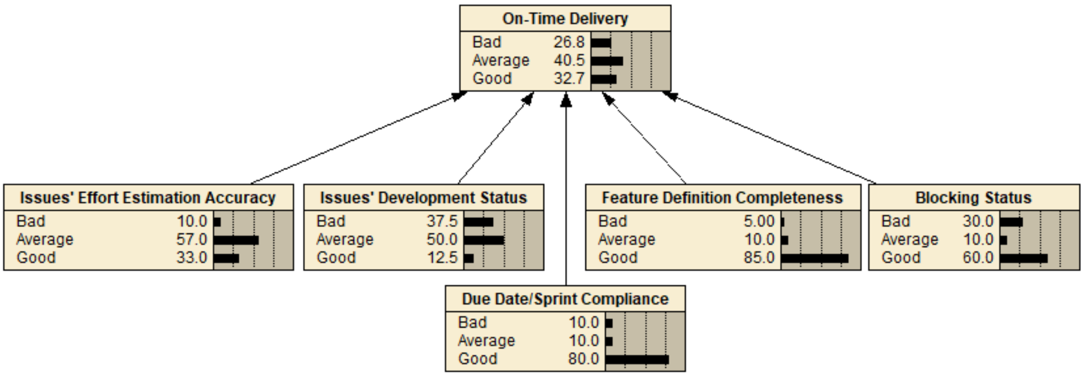


Figure 10 Excerpt of the estimation model for the *On-Time Delivery* SSI resulting from the internal methodology evaluation

Second, the *Product Quality* SSI was approached. Results and feedback gathered in such iteration were useful to formulate and polish the intended SESSI method. Details of this iteration were published and can be consulted in (Manzano et al., 2018b).

3.2.2.3 Stage 2-Iteration 3: Consolidated SESSI method

The third iteration focused on formulating a consolidated version of the SESSI method. So, we polished some aspects that we learnt from previous iterations in the context of the Q-Rapids partners. The main improvements were:

a) Based on the experience gained from the second iteration, we generalized and improved several aspects of the method and their intended resulting artifacts, including their guidance support.

b) We extended the method's tool support in order to provide data-driven capabilities and automate several steps of the method.

All in all, this iteration resulted in the consolidated version of the SESSI method presented as a result of this thesis and that was evaluated in Stage 3.

3.2.3 Stage 3-Solution Validation

It refers to the summative evaluation of the SESSI method in other industrial settings than those in which it was conceived. This stage was led by **RQ3** *"Is it feasible to apply* *the SESSI method to specify, estimate, and monitor SSIs for supporting evidence-based decision making in software companies?"*. In particular, we were interested in assessing the potential industrial worthiness and feasibility of the method as well as the perceptions from the practitioners about its applicability.

To provide a summative evaluation of the consolidated SESSI method, we applied the method in an industrial context different than the previous formative iterations. This evaluation was tackled as a case study performed in Modeliosoft, a subsidiary firm of Softeam. It focused on their *Product Readiness* SSI. Details on this case study were published in (Manzano et al., 2021) and are provided in Chapter 8.

It is important to remark that after the publication of (Manzano et al., 2021), we further improved the understandability of the SESSI method by making small changes in the explanation of some method activities and nomenclature used. The final version of the SESSI method is detailed in Chapters 4, 5, 6 and 7. The details of the summative evaluation are provided in Chapter 8.

3.2.4 Stage 4-SESSI's Usefulness Insights

It refers to the exploration of the potential benefits that the resulting assets from the SESSI method can bring to the software companies for improving their decision-making processes. This stage was led by **RQ4** *"Is it feasible to use the resulting assets from the SESSI method for enabling advanced decision-making support?".* We explored the feasibility of using the resulting assets from the SESSI method for fostering advanced decision-making support. In particular, we studied a company that previously applied the SESSI method and explored the feasibility of using the resulting models, data and infrastructure from the SESSI method for forecasting the values of relevant SSIs for covering their decision-making needs. This stage was based mainly on action-research and case study research. Details of the methodological approach and results are provided in Chapter 9.

Table 6 relates the research questions, objectives of the thesis, the research stages, and the main methodological approaches used in each stage. In addition, it states the chapters that provide detailed information.

Table 6 Correspondence between RQs, objectives, methodological approaches and chapters of the thesis

RQs	Objective	Research Stage	Main Methodological approach		Detailed in Chapter/s
RQ1	O1	Stage 1	Design Science	SLR	Chapter 2
RQ2	O1, O2, O3	Stage 2		Action-Research	Chapter 4-7
RQ3	O1, O2, O3	Stage 3		Case Study	Chapter 8
RQ4	O4	Stage 4	Action-Research		Chapter 9

4 Overview of the SESSI Method

This chapter provides a brief overview of the SESSI method so the reader may get familiar with the method. The following sections provide an overview of the three phases composing the method, while Chapters 5, 6 and 7 describe each of these phases in detail.

As mentioned in Chapter 3, the SESSI method resulted from multiple research iterations led by **RQ2** and aimed to deal with the three main problems identified in this thesis, introduced in Chapter 1, section 1.2.

The SESSI method allows software companies to specify, estimate, and monitor SSIs according to their informational needs and potential data availability, with the aim of providing evidence to support strategic decisions related to such SSIs. The method aims to be a helpful asset for software companies' roles related to strategic decisions. These roles mainly consist, for instance, of CEOs, CTOs and Product Owners (Aurum et al., 2006).

The SESSI method aims to be adopted and adapted by interested software companies according to their business context and needs. Domain experts from the companies should lead the execution of the SESSI method. The method implies some elicitation tasks aimed to gather domain knowledge from the company, as well as technical activities that are supported by software tools.

As shown in Figure 11, the SESSI method is composed of three phases. The first one corresponds to the specification of the SSI of interest according to the informational needs and the potential data availability of a software company. The second phase deals with the construction of the corresponding SSI estimation model to enable its assessment according to the company's criteria. Finally, the third phase deals with the deployment of the SSI estimation model on the company's premises to operationalize the continuous SSI monitoring thorough the software project lifecycle. Figure 11 shows a graphical overview of such three phases, including a high-level view of the inputs and outputs of each one.

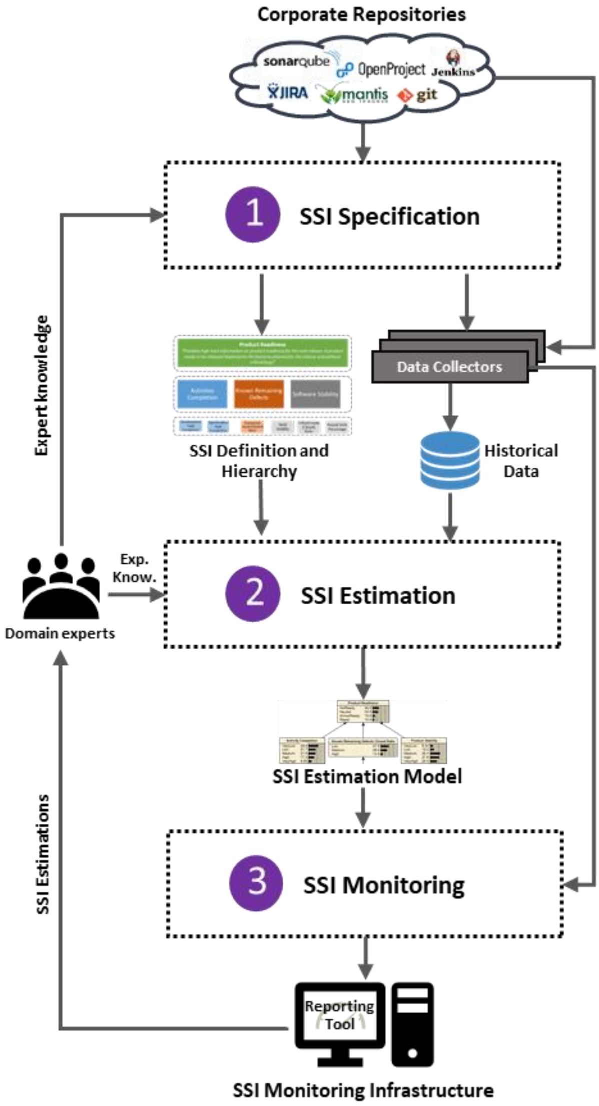


Figure 11 Overview of the SESSI phases with their inputs and outputs

In the following sections we provide an overview of the three phases of the method.

4.1 SSI Specification

The first phase of the SESSI method corresponds to the specification of the SSI of interest. This phase requires the participation of strategic, tactical, and operational-related roles that are related to the SSI's decision-making processes. It aims to specify SSIs in such a way that helps to overcome the following problems, already introduced in Chapter 1:

- Subjectivity and context dependency. SSIs are context-dependent and should fit the informational needs of each software company in order to be a useful support asset for decision-making processes. To deal with this, the specification phase of the SESSI method provides guidelines to support software companies to elicit and define the SSI meaning based on their specific context and needs and information available, or potentially available from their corporate repositories, including the development of data collectors to be used in the subsequent phases of the SESSI method.
- Focus on SSIs (i.e., evidence to support strategic decisions). Despite the existence of numerous literature proposals and project management tools providing operational indicators, many of them lack the focus on SSIs able to provide evidence to support strategic decisions. The SESSI method addresses this gap by considering the SSI specification as a suitable hierarchical structure composed of lower-level information that can be potentially gathered from corporate repositories, including expert knowledge.

Details of this phase are provided in Chapter 5.

4.2 SSI Estimation

The second phase of the SESSI method deals with the construction of the SSI estimation model to enable the assessment of the SSI specified in Phase 1. This phase requires the participation of strategic, tactical, and operational-related roles that are related to the SSI's decision-making processes.

The SSI estimation models built in this phase are based on Bayesian Networks (BNs), a type of probabilistic, graphical models able to cope with the problems related to the SSI estimation previously listed in Chapter 1:

- Combining corporate repositories data exploitation and expert knowledge. Not only the SSI specification is subjective to the software company's needs and requirements, but the decision-making process is also subjective: strategic decisions are conditioned by numerous factors. Thus, the SSI estimation model needs to combine data and expert knowledge in order to be able to properly inform decisions. The Bayesian nature of the SSI estimation model built in this phase allows modeling the subjective relationships among the hierarchical components representing the SSI, considering not only data points but also expert knowledge.
- Dealing with lack of data/information. Strategic decisions are, unlike operational decisions, non-routine decisions, which are normally taken in contexts of incomplete data/information (i.e., influenced by unknown and/or unobservable external factors, partial lack of the input data required to estimate the SSI from the corporate repositories, etc.), which happens not only in software engineering but in other contexts (Johnson et al., 2007). The SSI estimation models from the SESSI method, are able to deal with this by modeling and representing uncertainty through probabilistic relationships.
- Explainability and understandability. Despite the importance of providing a comprehensive connection between low-level data used as evidence and the higher-level information needs of strategic decisions, studies show a lack of support for this aspect (Cito, 2016; Figalist et al., 2021; Martinez-Fernandez et al., 2019; Mesquida Calafat et al., 2022). This can support decision makers for making better informed decisions (Dam et al., 2018; Figalist et al., 2021). SSIs estimation models are conceived as BN models that keep track of the connection among the different levels of the hierarchy that compose the SSI providing an intuitive and visual representation that favors explainability and understandability (in contrast to black box models such as neural networks). Therefore, SSIs estimation models can provide drill-down capabilities down to the evidence data.

Details of this phase are provided in Chapter 6.

4.3 SSI Monitoring

The third and last phase of the method relies on the previous ones and aims to put forward the required infrastructure for enabling the SSI monitoring. Such monitoring could be useful for any role related to the SSI's decision-making processes.

This phase deals with the related aspects for deploying and using SSI estimation models as the basis for their monitoring, including the main elements needed to provide a monitoring infrastructure. It also suggests a generic architecture that has been successfully used in the context of the Q-Rapids project.

Details of this phase, including the suggested elements and architecture for putting forward the SSI monitoring are provided in Chapter 7.

A detailed description of the three SESSI method phases is provided in Chapters 5, 6 and 7. To illustrate such three phases of the SESSI method along such chapters, we will use an SSI to estimate the *Product Quality* SSI. This SSI is a modified version of the one presented in a previous work (Manzano et al., 2018b), resulting from conducting a preliminary, formative evaluation of the SESSI method with one of the Q-Rapids industrial partners (see Chapter 3, section 3.2.2.2).

5 SESSI Phase 1: SSI Specification

This chapter describes the first phase of the SESSI method, which aims to obtain the definition and specification of the SSI of interest.

The SSI specification phase of the SESSI method consists in three steps aimed to obtain 3 relevant assets:

- An SSI textual definition.
- An SSI hierarchical specification.
- Data collectors.

The following subsections detail these assets and the techniques considered in the SESSI method for eliciting/obtaining such assets.

To illustrate the steps of this phase, we will present excerpts from the specification of the *Product Quality* SSI in the context of a software development project from a Q-Rapids partner.

5.1 SSI Textual Definition

The SSI textual definition aims to provide a high-level overview of the SSI and it is used as the basis to further inquiry on the SSI and decomposing it into more detailed aspects.

The SSI textual definition should be stated and agreed by the roles related to the decision-making processes around the SSI (i.e., strategic-related roles). It is important to remark that the provided definition should reconcile the tacit knowledge from the roles related to the SSI and the data available in corporate repositories. It is, while the SSI definition should embrace a high-level overview, it should be kept in mind that it will be further decomposed into lower-level measurable components (i.e., existing data already collected in corporate repositories or that can be or potentially collected).

The very initial definition of the *Product Quality* SSI was provided by a Q-Rapids partner. It aims to enable the overall quality assessment of a determined software product under development, therefore supporting decisions related to its development process and planned release. These decisions include taking actions related to specific software development aspects that may negatively impact the overall product quality, and ensuring the software product under development is released while meeting its quality requirements.

To deep and confirm such definition, the role related to the *Product Quality* SSI was approached. This role was the Product Owner of the product under development. This role oversees the decisions related to the product management, in order to ensure that the clients' needs, and requirements (including quality requirements) are met. Therefore, the Product Owner was interested in having supporting evidence for assessing the overall quality of the software product and taking specific decisions regarding the aspects conforming such quality. According to the domain knowledge (including company's rules) from the Product Owner, the quality of the software product under development is composed of its stability, the testing status, and the codebase quality (in order to ease maintenance tasks and prevent future issues such as code smells and security vulnerabilities). Therefore, the Product Owner stated the SSI textual definition as the "Degree of fulfilment of the quality requirements for the product, including aspects related to its codebase quality, its testing status, and overall stability of the product" (see Figure 12 or Table 7). The *Product Quality* SSI should be able to provide evidence on the quality status of the software product under development and enable decisions such as postponing the product release due to not meeting their quality criteria, while enabling taking actions on specific, lower-level aspects of such quality criteria (such as planning a codebase refactoring due to the presence of code smells).

5.2 SSI Hierarchical Decomposition

To decompose the SSI from its definition (specified in the previous step), the SESSI method considers the hierarchical decomposition (i.e., breakdown) of the SSI in terms of three types of elements from the Quality Model (QM) structure proposed in (Franch et al., 2017; Martinez-Fernandez et al., 2018), which is at the same time based on hierarchical elements of the Quamoco approach (Wagner et al., 2015) and has proved to contribute to ease the specification and understandability of software development-related concepts.

The three levels of the hierarchy are as follows:

- **Metrics** refer to lowest-level, operational aspects of the product or development process that may be directly extracted or computed from raw measures from the corporate repositories of the software company (e.g., project management tools such as Jira, SonarQube, Jenkins, Git, logs, user feedback, domain information stored in databases, etc.). The roles that should specify or confirm the metrics are usually operational roles involved in the software development process, such as software developers, Continuous Integration/Continuous Delivery (CI/CD) developers, Quality Assurance (QA) developers, etc. In the *Product Quality* SSI example, the metrics and their rationale are specified by these roles in collaboration with the tactical roles involved in the factors' specification (see next bullet). Some examples of the specified metrics are *Code Complexity*, which is computed using the raw measures *# lines of code* and *# non-duplicated lines of code* from SonarQube, and *% Passed Integration Tests*, computed from raw measures # *integration tests passed*, and # *integration tests ran* from Jenkins. The complete list of metrics for the *Product Quality* SSI example is shown in Figure 12, while their rationale is presented in Table 7.
- **Factors** refer to aggregations of data providing meaningful information about the SSI. The SESSI method considers the factors as aggregations of metrics and/or other factors. Factors should be specified by the roles involved in decision making at the tactical level (i.e., Project Managers, QA lead, CI/CD lead, etc.) in conjunction with the roles directly concerning the SSI. In the case of the *Product Quality* SSI used as example, its specified factors are *Code Quality*, *Testing Status* and *Software*

Stability. The factors of the *Product Quality* SSI example are graphically shown in Figure 12, while their definition is presented in Table 7.

- **SSI** refer to the top level of the hierarchy that relies on the previously defined factors. It represents aspects or characteristics related to software products and/or software development processes that a software company considers strategic or important for their decision-making processes.

To assist the obtention of the SSI hierarchical decomposition, we suggest the use of some information sources and elicitation techniques (Lethbridge et al., 2005):

- **Literature reviews:** To support the specification of the SSI, one could consult some literature that provide insights on the relevant aspects of the SSI. This could help as starting point for the textual definition or the hierarchical decomposition of the SSI. We have found that several product and process-related SSIs such as *Software Readiness, Product Quality* or *Process Performance* have been proposed in the literature (Antolić, 2008; Meidan et al., 2018; Staron et al., 2014, 2012; Wagner et al., 2015). Authors usually define these SSIs based on case studies conducted in industrial environments or using data from open-source software development projects. So, some ideas might raise from the assessment of such literature.
- **Interviews:** The use of semi-structured interviews (specifically designed for each corresponding role) with the aim to elicit information to decompose the high-level definition of the SSI into factors and metrics was a helpful instrument for supporting the SSI hierarchical decomposition.
- **Workshops:** To assist on the aggregation of meaningful information for the company purposes, we propose in-situ workshops with diverse roles related to the SSI as participants for the identification of factors and metrics. In particular, approaches such as the Goal-Question-Metric (GQM) (Solingen et al., 2002) or the GQM+Strategies (Basili et al., 2010) might be useful for driving the interaction of the participants of the workshop (Basili et al., 2007). On the one hand, the GQM approach may support the identification of metrics for specific factors (i.e., to decompose a hypothetical *Code Quality* factor, the question "what is code quality?" may lead to the identification of metrics related to the code size, complexity, and defects). On the other hand, GQM+Strategies may potentially support the overall connection between the different components of

the hierarchical decomposition, as it is an approach meant to link goals and strategies across organizational levels basing on measurement. These approaches were successfully used during the Q-Rapids project with industry partners to support the hierarchical decomposition of SSIs such as *Product Quality* and *On-Time Delivery* (Q-Rapids, 2019a, 2018b).

We used these three types of elicitation techniques, i.e., we used literature reviews complemented with interviews and workshops. The roles involved in the interviews and workshops were those related with the SSI's decision-making processes: i.e., strategic-related roles such as CEOs, CTOs and Product Owners; tactical-related roles such as Project Managers, QA leaders; as well as operational-related roles such as developers or testers.

To decompose SSIs, we usually initiated by collecting a list of candidate factors (that should be specific and quantifiable) from the literature review and/or interviews with strategic and tactical-related roles. Then, during the workshops, we elaborated such information as follows: For each factor, a set of one or more metrics were identified by the workshop's participants in order to measure the factor. As metrics will be computed from data directly coming from the corporate repository tools, we highlighted the importance of suggesting objective and quantitative metrics. This process was mainly performed by tactical-related roles with the help of operational-related roles.

Subsequently, the set of metrics were mapped to the corresponding corporate tools that can derive them. It included the specification of the mappings and/or raw data aggregations for deriving the expected metrics. This process was mainly performed by operational-related roles.

The graphical specification of the *Product Quality* SSI is depicted in Figure 12, while the detailed specification, including the rationale of the SSI components is shown in Table 7.

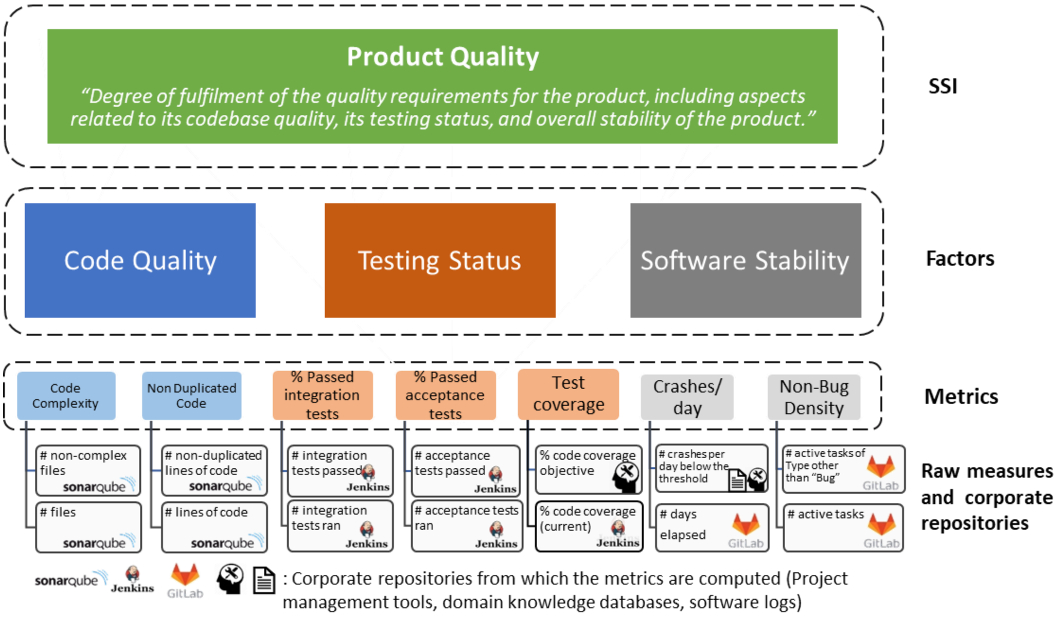

Product Quality
"Degree of fulfilment of the quality requirements for the product, including aspects related to its codebase quality, its testing status, and overall stability of the product."
SSI
Code Quality
Testing Status
Software Stability
Factors
Code Complexity
Non Duplicated Code
% Passed integration tests
% Passed acceptance tests
Test coverage
Crashes/ day
Non-Bug Density
Metrics
# non-complex files
# files
# non-duplicated lines of code
# lines of code
# integration tests passed
# integration tests ran
# acceptance tests passed
# acceptance tests ran
% code coverage objective
% code coverage (current)
# crashes per day below the threshold
# days elapsed
# active tasks of Type other than "Bug"
# active tasks
sonarqube
Jenkins
GitLab
Raw measures and corporate repositories
: Corporate repositories from which the metrics are computed (Project management tools, domain knowledge databases, software logs)

Figure 12 Graphical summary *Product Quality* SSI specification

5.3 SSI Data Collectors

Data collectors refer to software artifacts developed with the purpose of enabling the automatic gathering, computation, and storage of the metrics from corporate repositories.

Data collectors are a crucial artifact for SSI monitoring as they collect input data required to automatically estimate the SSIs. In addition, data collectors also support the construction of the SSI estimation models that allows such monitoring as it will be explained in subsequent chapters.

Software companies can develop their own data collectors, reuse, or customize them from other projects, according to their architectural and programming language requirements.

In the context of the Q-Rapids project, some open-source data collectors were developed by the academic and industrial partners. The developed data collectors included support for project management tools such as Jira, Jenkins, OpenProject, GitHub, and SonarQube, among others. These connectors were developed as modular extensions of an integrated Java open-source software library available in GitHub[16].

[16] Qrapids-connect (https://git.io/JvGwf)

Such connectors can be freely adopted and modified by companies interested in conducting the SESSI method.

Table 7 Specification of the *Product Quality* example SSI

Textual Definition: *Degree of fulfilment of the quality requirements for the product, including aspects related to its codebase quality, its testing status, and overall stability of the product.*					
Factor	**Description**	**Metric**	**Description**	**Data Source**	**Definition** (In some cases, the metric's definition shown is simplified)
Code Quality	Measures the quality of the source code through static code analysis-related metrics	Code Complexity	Ratio of non-complex source code files with respect to the total number of source code files	SonarQube	$\frac{\#\ \mathit{non\ complex\ source\ code\ files}}{\#\ \mathit{source\ code\ files}}$
		Non-Duplicated code	Percentage of non-duplicated code with respect to the total number of lines of code	SonarQube	$\frac{\#\ \mathit{non\ duplicated\ lines\ of\ code}}{\#\ \mathit{lines\ of\ code}}$
Testing Status	Summarizes the tests performed by the QA team to make sure that the criteria and thresholds agreed are met and that the system performs as specified	% Passed integration tests	Percentage of successful integration tests with respect to the total of integration tests triggered in a specified period	Jenkins	$\frac{\#\ \mathit{integration\ tests\ passed}}{\#\ \mathit{integration\ tests\ ran}}$
		% Passed acceptance tests	Percentage of successful acceptance tests with respect to the total of acceptance tests triggered in a specified period	Jenkins	$\frac{\#\ \mathit{acceptance\ tests\ passed}}{\#\ \mathit{acceptance\ tests\ ran}}$
		Test coverage	Measures the degree to which the source code is executed when a particular test suite is executed	Jenkins, expert knowledge	$\frac{\%\ \mathit{current\ code\ coverage}}{\%\ \mathit{objective\ of\ code\ coverage}}$
Software Stability	Measures the status of the operational software quality of the monitored release, taking into consideration the density of bugs and the crashes/day	Crashes/day	Percentage of days with crashes below the specified threshold	Logs, GitLab	$\frac{\#\ \mathit{days\ with\ crashes\ below\ the\ threshold}}{\#\ \mathit{days\ elapsed\ since\ the\ develop.\ started}}$
		Non-Bug Density	Ratio of tasks classified as other than "bug" with respect to the total number of open issues	GitLab	$\frac{\#\ \mathit{active\ tasks\ of\ type\ other\ than\ "bug"}}{\#\ \mathit{active\ tasks}}$

6 SESSI Phase 2: SSI Estimation

The second phase of the SESSI method aims to enable the SSI assessment through the construction of an SSI estimation model that is based on Bayesian Networks (BNs).

The use of BN properties helps to: a) deal with the complexity of representing strategic aspects in the presence of uncertainty and other affecting factors; b) include experts knowledge in the estimation model, c) infer probabilities even if some data is missing; and d) ensure the explainability of the SSI estimation model and its inferences when used for the SSI monitoring.

This chapter presents a background on BN, the selected technique to build the SSI estimation models, followed by the detailed description of the method phase.

6.1 Bayesian Networks

Bayesian belief networks, or simply BNs are a type of probabilistic, graphical models able to represent causal relationships between a set of modelled continuous or discrete variables. BNs can be built from data and/or domain expertise, and they can be used for a wide range of applications such as reasoning, prescriptive analysis, anomaly detection, and prediction. Being probabilistic and graphical models, BNs incorporate parts of probability and graph theories.

Probabilistic inference in BNs is based on the Bayes theorem (Zhang et al., 2008), which can be easily inferred from the axioms of conditional probability. The graphical part of BNs consists in a Directed Acyclic Graph (DAG) with the form *(Vertices, Edges)* or simply *(V, E)*, representing the joint probability distribution over the set of the considered random variables, represented by the set of nodes (i.e., nodes in BN terminology, vertices in graph terminology) *V*. Conditional dependence between such variables is denoted by directed edges belonging to the *E* set. Depending on the nature of the variables represented by the BN nodes, these may be continuous or discrete. Nodes with continuous variables are parameterized using probability functions, and nodes with discrete variables (i.e., nominal, ordinal, interval, ratio, or discrete numbers) using Conditional Probability Tables (CPTs) specifying their probability distribution. Therefore, the joint probability distribution can be computed through the chain rule, applying the Bayes theorem over individual node probabilities.

BNs can be constructed from data and expert knowledge. When there is enough data available of every variable to be modelled, both the BN structure and its CPTs can be learnt automatically from data. However, in practice, this situation is not common and expert knowledge is needed to build, or complete the BN. Even in situations with enough data, such expert knowledge can provide information like key relationships that data alone would fail to discover (Constantinou and Fenton, 2017). However, for large BNs, the manual elicitation becomes impractical, as each node’s CPTs grows exponentially based on the number of parent nodes and their potential states. Therefore, its manual specification would be prohibitive in terms of required time and effort.

To reduce the elicitation burden and the domain experts’ fatigue and overwhelm in those scenarios, the literature has proposed several elicitation methods, heuristics, and specific approaches (“Noisy-OR” and the “Noisy-MAX” (Kincaid and Cheney, 2002), “Ranked Nodes” method (Fenton and Neil, 2005; Fenton et al., 2007) and the Weighted Sum Algorithm (WSA) technique (Das, 2004)). Although the "Ranked Nodes" is the most popular CPT's probability elicitation method, it has some limiting assumptions (nodes' states need to be ordinal, the assumption of the underlying node's Truncated Normal as its probability distribution, the need to specify statistical parameters for such distribution and the limited range of mixture functions among the parent nodes towards the child node under quantification). Such assumptions make the WSA a feasible technique when any of these cannot be satisfied, which is our case (we don't assume neither the nodes' underlying distribution, ordinal states, or the statistical knowledge by the domain experts).

The WSA technique is expert-based and tackles the probabilities' elicitation problem with the foundation of the "compatible configurations". These compatible configurations refer to combinations of parental states (of the child node under quantification) that make more sense, and/or can coexist more often according to the domain experts. To use the WSA, the domain experts need to provide such configurations along with the resulting child node's state probabilities. Additionally, they need to specify the relative weights quantifying the importance of each parent node towards this child node under CPT's quantification. Using this information, the WSA infers the complete child node's CPT by interpolation. Therefore, to derive the node's CPT, the domain experts only need to provide the subset of the CPT corresponding to the so-called compatible configurations and the relative weights.

6.2 Building the SSI Estimation Model

The inputs of this phase are: the assets from the SSI specification phase, expert knowledge and historical data collected from the corporate repositories through data collectors. Figure 13 shows the steps required to build an SSI estimation model based on BNs. Each one of these steps are further detailed in the remaining of this chapter.

To support this phase, a set of supporting tools was developed. These tools are introduced in this chapter and are further described in the Appendix 1.

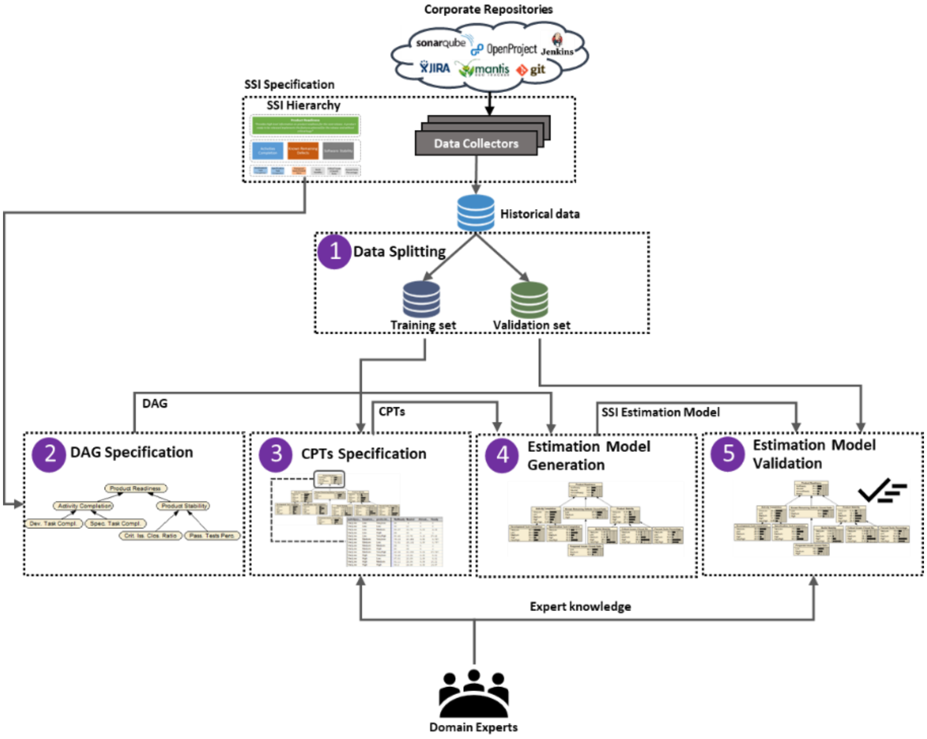


Figure 13 Overview of the SESSI steps to build the SSI estimation model

6.2.1 Data Splitting

This first step aims at generating training and validation sets from the historical data collected with the data collectors resulting from the SSI Specification phase (i.e., metrics values). The objective of splitting the dataset into training and validation sets is to build the SSI estimation model using the training set, followed by the evaluation of the predictive performance of the resulting SSI estimation model using the validation set.

Splitting the dataset is a crucial step to assess the generalization capacity of any estimation/prediction model (Raschka, 2018). It is recommended to split the available historical data in commonly used training/validation splits such as 70/30% or 80/20% respectively, or even 90/10% if the amount of historical data is relatively large (Raschka, 2018). It is important to remark that the accuracy of the resulting SSI estimation model will be influenced by the amount and quality of the data used.

Lastly, one consideration should be taken when splitting the data in this step: when dividing the historical data into the training and validation sets, it is recommended to maintain the original distribution of the dataset. It is because preserving the independence and identically distribution of the training set is important when building any kind of predictive model (Raschka, 2018).

6.2.2 DAG Specification

As mentioned above, the construction of the SSI estimation model is based on BNs. The graphical structure of a BN is determined by a DAG.

This step aims to specify the DAG of the SSI estimation model, based on the hierarchical decomposition determined in the previous SSI specification phase. Thus, the nodes of the DAG will correspond to the elements of the SSI specification hierarchy, (i.e., SSI, factors, and metrics) together with the established edges in the SSI specification hierarchy (i.e., the direction of an edge specifies which node is impacted (child node, edge's target) by which node (parent node, edge's origin). Metrics are represented in the BN by nodes yielding at the bottom level of the DAG (i.e., metric nodes), while factors are represented by intermediate nodes of the DAG. The top-most level of the DAG represents the SSI.

The most relevant elements for the intended SSI estimation model are metrics. Its importance resides on the fact that their values (gathered through data collectors) will constitute the input of the SSI estimation model once it is built and used for monitoring and assessing the SSI. In other words, the values of the metrics will propagate upwards the DAG through belief propagation to infer the probabilities of factors and SSI nodes.

Figure 14 shows a DAG example of the BN estimation model for the *Product Quality* SSI example. Such estimation model is composed of 3 factor nodes and 7 metric nodes, plus the SSI itself. It can be observed the metric nodes that impact the factor nodes, thus establishing a cause-effect relationships. The three factor nodes, at their turn, impact directly on the SSI.

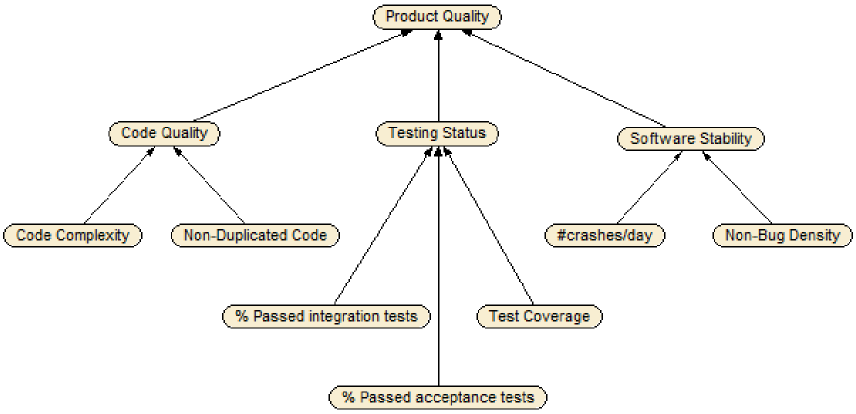


Figure 14 Example DAG for the *Product Quality* SSI estimation model

6.2.3 CPTs Specification

In order to build the BN, a CPT should be generated for each node of the DAG. The CPT of a node defines the conditional probabilities of each possible state (value) of such node with respect to the states (values) of their parent nodes, if any. The definition of the states and the probabilities allows incorporating expert knowledge to the resulting model.

To build the CPTs, one should first specify the states for each of the DAG nodes. This should be done according to the domain knowledge (provided by domain experts) and the needs of the decision makers concerning each node. Examples of such states are nominal states such as {*True, False*}, {*Ready, Not Ready*}, or ordinal, such as {*Low, Medium*, *High*}. For the metric nodes, which do not have any parent node and are directly computed from the metrics values, it is necessary to specify a binning function (also known as discretization function) for each of these nodes, to translate the metrics values from the corporate repositories into the BN corresponding discrete state. Such binning functions are mainly required by domain experts as they find that quantifying the information of each CPT using discrete states is easier than using continuous values, as also found by (Chen et al., 2017; Halford, 2022). In addition, the definition of discrete states allows domain experts to denote semantics into the labels of the states at their convenience (e.g., bad-good, low-high, not ready-ready). For example, in the *Product Quality* example BN, the metric node *Test Coverage* may have three ordinal states defined by the Quality Assurance (QA) team as {*Low, Medium, High*}. Such metric may be computed by the data collectors in the continuous, numeric [0,1] interval, and discretized using binning intervals defined as [0-0.5) for the “Low” state, [0.5-0.75) for “Medium” and [0.75-1] for “High”.

As support for the manual specification of the binning intervals by the domain experts, some unsupervised binning methods can be used as a starting point. Unsupervised binning methods transform numerical variables into discrete ones without relying in target or class information (ground truth). The two main unsupervised binning methods are the Equal-Width and Equal-Frequency (Dougherty et al., 1995). We provide software artifacts implementing versions of these algorithms as part of the SESSI tool support. For more details, check Table 8 and the Appendix 1 (section 1). The intervals obtained with the use of such software artifacts can be used directly “as it is”, or as a basis when the domain experts specify the binning intervals according to their needs and/or domain rules.

Once the states of each node have been defined, the process to fill the CPT for each node is performed differently according to the type of the node. We differentiate the cases of metric nodes and factor and SSI nodes. The process to fill the CPTs for these cases is explained as follows:

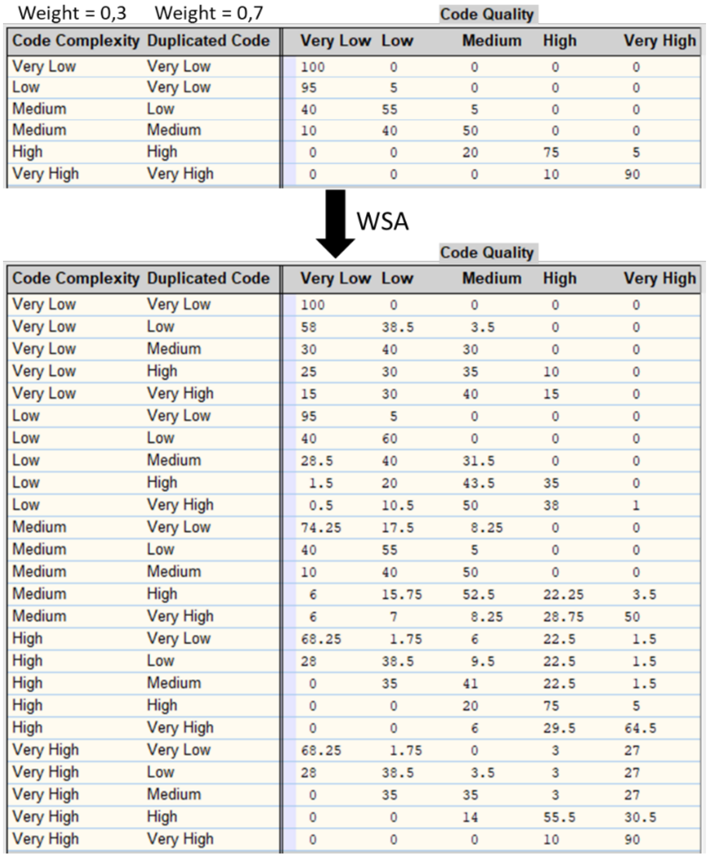
- **Metric nodes:** Being at the root level of the BN, these nodes do not have any other parent nodes impacting them, therefore their CPTs are not conditioned by other nodes states. The probabilities to fill in their CPTs only depend on their specified states. The SESSI method requires the use of frequency quantification over the training set, in order to automatically quantify the CPTs for these nodes For this purpose, the required information consists of a) the training set for the metric nodes under quantification, b) the states specified for each metric node under quantification, and c) the specified binning intervals for each one. The frequency quantification is performed by binning the numerical training data of the metrics into the elicited states, and then computing the proportion of data yielding in each state with respect to the total.

  We have implemented a software artifact that performs such quantification automatically using the provided historical data (i.e., *GetFrequencyQuantification*). For more details, check Table 8 and the Appendix 1 (section 2). The frequency quantification results can be used as a basis to determine the probabilities for the CPTs of the metric nodes under quantification. These probabilities to fill the CPT in should reflect the realistic state/knowledge about the node under quantification before any evidence is entered into the BN model. There are cases in which the automatically computed probabilities may need manual refinement by the domain experts. We illustrate this probability refinement process with an example extracted from the *Product Quality* SSI. In this case, the frequency quantification for the node *Bug Density* using a training set composed of the 70% of the historical data may result in the quantified probabilities {Very Low = 35.3%, Low = 53.4%, Medium = 11.3%, High = 0%, Very High = 0%}. However, in this case, the development and QA team adjusted the probabilities when entering them into the node's CPT, resulting in {Very Low = 35.3%, Low = 53.4%, Medium = 6%, High = 3.25%, Very High = 2%}, as specifying 0% probabilities for the states "High" and "Very High" would cause the model to not admit any input evidence of the node being in such states, as per the 0% chance of happening.

- **SSI and factor nodes:** CPTs for these child nodes (as they have other nodes impacting them) are conditional on the states of their parent nodes. For child nodes with only a parent node, the CPT filling process can be performed using the parent node CPT as a basis. Although these cases pose the possibility of grouping the two nodes in one ("node absorbing" in BN terminology), having both separate nodes can be useful to add uncertainty to the relationship between both child and parent node. In contrast, for child nodes with more than a parent node, CPTs are conditional on each possible combination of its parent nodes states. Therefore, the size of these CPTs grows exponentially on the number of parent nodes and their states. Therefore, it might be too exhausting or not realistically feasible for the domain experts to fill these CPTs manually, as it can involve specifying hundreds or thousands of probabilities. Hence, to address these cases, we suggest the use of techniques and tools to ease the CPT elicitation process.

  Specifically, the SESSI method considers the use of our developed implementation of the WSA (Das, 2004), given its strengths compared to other common techniques such as the Ranked Nodes method (see section 6.1 for more details). We implemented our version of the WSA in Java, as open-source software and freely available to potential interested users. For more details, check Table 8 and the Appendix 1 (section 2). The WSA was conceived as an expert-based technique to ease the probability elicitation process. It takes advantage of the availability and simulation heuristics, taking as input a subset of the CPT (i.e., compatible configurations) and the relative weights of the parent nodes towards the child node under quantification. The subset of the CPT to determine corresponds to the mentioned compatible (parental) configurations that are more prone to happen, according to the domain expertise. We have implemented two software artifacts able to determine such compatible configurations automatically from the training data, i.e., *GetCompatibleConfigurations* and *GetChildCompatibleConfigurations*. For more details on these software artifacts, check Table 8 and the Appendix 1 (section 2). Taking this subset of the CPT as input, along with the relative weights of the parent nodes, the WSA infers the complete CPT automatically. For the cases in which the domain experts struggle to determine the relative weights of the parent nodes towards the child node under quantification, we suggest the use of the Analytic Hierarchical Process (AHP) through an existing online implementation[17] (Goepel, 2018). The AHP can determine the weights automatically by asking the domain experts a set of pairwise comparisons between the parent nodes towards the child node. An example of the WSA required input for the *Code Quality* node, part of the *Product Quality* example SSI is shown on Figure 15. It shows the application of the WSA technique for such node. The required input by the WSA consists of a set of compatible configurations (one per each parent node state), the corresponding resulting child node probabilities for each one, and finally the relative weights of the parent nodes toward the child node *Code Quality*, which were elicited through the AHP method previously mentioned. In this case, the WSA inferred the complete CPT, which contained a total of $5^2 = 25$ rows of 5 probabilities each one.

[17] https://bpmsg.com/ahp-online-calculator/



Weight = 0,3	Weight = 0,7	Code Quality				
Code Complexity	**Duplicated Code**	**Very Low**	**Low**	**Medium**	**High**	**Very High**
Very Low	Very Low	100	0	0	0	0
Low	Very Low	95	5	0	0	0
Medium	Low	40	55	5	0	0
Medium	Medium	10	40	50	0	0
High	High	0	0	20	75	5
Very High	Very High	0	0	0	10	90

		Code Quality				
Code Complexity	**Duplicated Code**	**Very Low**	**Low**	**Medium**	**High**	**Very High**
Very Low	Very Low	100	0	0	0	0
Very Low	Low	58	38.5	3.5	0	0
Very Low	Medium	30	40	30	0	0
Very Low	High	25	30	35	10	0
Very Low	Very High	15	30	40	15	0
Low	Very Low	95	5	0	0	0
Low	Low	40	60	0	0	0
Low	Medium	28.5	40	31.5	0	0
Low	High	1.5	20	43.5	35	0
Low	Very High	0.5	10.5	50	38	1
Medium	Very Low	74.25	17.5	8.25	0	0
Medium	Low	40	55	5	0	0
Medium	Medium	10	40	50	0	0
Medium	High	6	15.75	52.5	22.25	3.5
Medium	Very High	6	7	8.25	28.75	50
High	Very Low	68.25	1.75	6	22.5	1.5
High	Low	28	38.5	9.5	22.5	1.5
High	Medium	0	35	41	22.5	1.5
High	High	0	0	20	75	5
High	Very High	0	0	6	29.5	64.5
Very High	Very Low	68.25	1.75	0	3	27
Very High	Low	28	38.5	3.5	3	27
Very High	Medium	0	35	35	3	27
Very High	High	0	0	14	55.5	30.5
Very High	Very High	0	0	0	10	90

Figure 15 Excerpt of the input data required by the WSA (top) for the *Code Quality* node and inferred CPT after the WSA application (bottom)

6.2.4 Estimation Model Generation

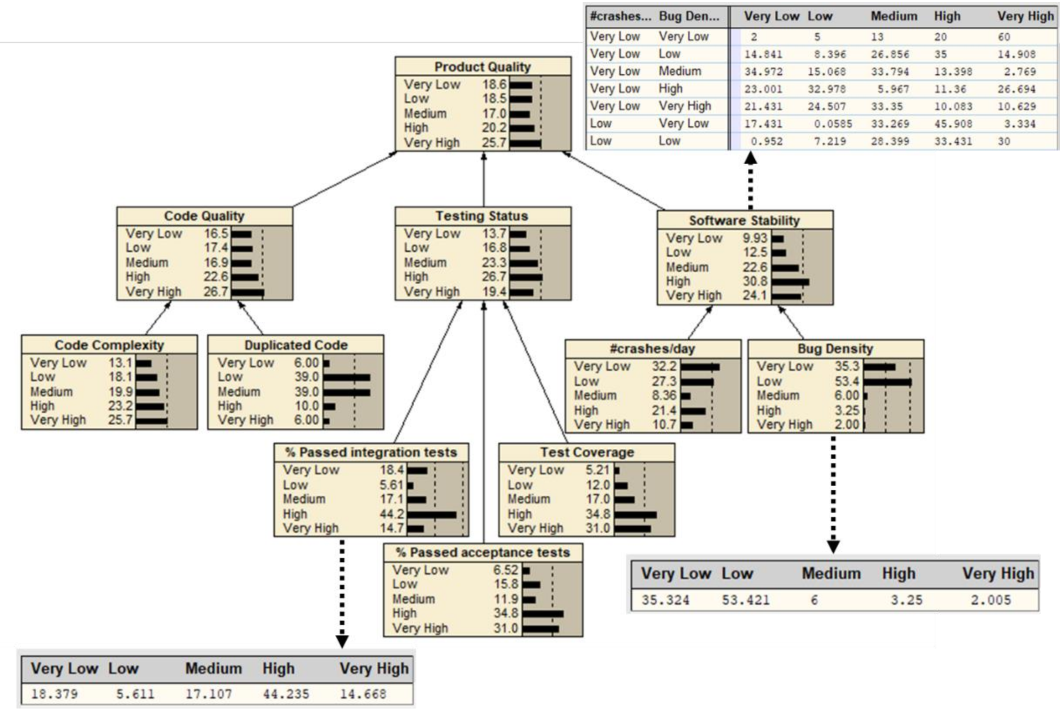
This step comprises the BN estimation model building. For this process, we consider the use of existing BN tools such as Netica®[18] and unBBayes[19]. The nodes are represented as discrete nodes in the BN, and edges representing conditional relationships are added between them as described in the DAG Specification step (section 6.2.2). We chose these tools as they provide a graphical UI, which is very useful to take advantage of the graphical and interpretability aspects of BNs, engaging domain experts to see how their knowledge is embedded into the SSI estimation model. Furthermore, these tools enable the SSI to be graphically assessed through "what-if" analysis performed directly with the estimation model. Additionally, these tools offer

[18] https://www.norsys.com/netica.html

[19] http://unbbayes.sourceforge.net/

APIs to enable the use of the BN models programmatically, either for their construction/modification, and for their assessment, thus enabling the SSI monitoring through external tools such as dashboards using such APIs.

As an example, to illustrate the result of this step, the assembled *Product Quality* SSI estimation model is shown in Figure 16. It shows the specified DAG and the default probabilities of each node before entering any evidence into the model, i.e., the *prior* probabilities computed from the CPTs. For metric nodes, these probabilities coincide with their specified CPTs (shown in Figure 16 for *% Passed Integration Tests* and *Bug Density* nodes), as they are not conditionally impacted by other nodes. Additionally, a partial CPT for the *Software Stability* factor node is shown, which is a child node whose CPT is conditionally dependent on two metric nodes.



#crashes...	Bug Den...	Very Low	Low	Medium	High	Very High
Very Low	Very Low	2	5	13	20	60
Very Low	Low	14.841	8.396	26.856	35	14.908
Very Low	Medium	34.972	15.068	33.794	13.398	2.769
Very Low	High	23.001	32.978	5.967	11.36	26.694
Very Low	Very High	21.431	24.507	33.35	10.083	10.629
Low	Very Low	17.431	0.0585	33.269	45.908	3.334
Low	Low	0.952	7.219	28.399	33.431	30

Very Low	Low	Medium	High	Very High
35.324	53.421	6	3.25	2.005

Very Low	Low	Medium	High	Very High
18.379	5.611	17.107	44.235	14.668

Figure 16 Example of the obtained BN for the *Product Quality* SSI

6.2.5 Estimation Model Validation

The objective of the model validation step is to evaluate the resulting model from the previous steps, assessing its accuracy and recalibrating it, if necessary, before it is used in production settings for supporting evidence-based decision-making processes. The recalibration is performed by tuning the CPTs of the BN nodes that require changes. Two generally-used validation methods are considered for the purpose of this step, namely Model Walkthrough and Outcome Adequacy validations (Mendes, 2014; Mendes et al., 2018). These methods consider the validation for the model “global” target or response variable, which would correspond to the SSI node in the context of the SESSI method. However, we propose to conduct the validation process not only for the final response variable (SSI node), but for the rest of child nodes. This extension is particularly relevant to ensure the trustworthiness of the entire model, especially because of the use of the semi-automatic approaches employed to obtain and infer the CPTs for child nodes. For the metric nodes, it is not necessary to conduct such validation, as their CPTs are directly quantified from the training data and refined by the domain experts.

The two considered validation methods are described as follows. They are meant to be performed sequentially, as they are based on different strengths of evidence.

6.2.5.1 Model Walkthrough Validation

The Model Walkthrough is used in different fields such as medicine, decision analytic modelling and social sciences (Mendes et al., 2018). Model Walkthrough allows assessing the accuracy of the model subjectively, at face value. Therefore, the domain experts get an idea of how accurate the model is, according to their own beliefs. Comparing with the domain experts’ own beliefs is convenient in this type of validation as there is usually no ground truth data for SSIs and factors. For each child node to be validated, the validation process using Model Walkthrough is performed through the specification of a set of hypothetical (“what-if”) scenarios by the domain experts. For each scenario, the domain experts also specify their perceived resulting state for that child node under validation, i.e., the state that would yield highest probability. To cover as many parts of the CPT as possible, it is recommended that the domain experts specify diverse scenarios in terms of the parent nodes states. The domain experts do not need to specify probabilities, just their perceived most probable resulting states for the child node under validation. After specifying the what-if scenarios and the corresponding expected most probable state, the scenarios are introduced into the SSI estimation model as evidence.

The objective is to compare the BN inferred output, i.e., the most probable state from the child node under validation, to the resulting state perceived by the domain experts.

If there is a mismatch between the BN output and the domain experts' perception, the estimation model should be then recalibrated by tuning the child node CPT. Concretely, the CPT row corresponding to the scenario and the corresponding child node’s probabilities. In these cases, it is also recommended to revise and recalibrate (if necessary) the CPT rows immediately above and below the recalibrated one, as for similarity they may also need some probability tuning. After the recalibration, the validation can proceed until all the mismatches are amended.

The number of scenarios to use in this validation will influence the accuracy of the model. However, such number depends on the size of the CPT of the node under validation, likewise determined by the number of parent nodes and their states. From our experience, we recommend specifying a set of scenarios corresponding to 10%-30% of the child node CPT size (i.e., its number of rows), taking into account that larger percentages can have a positive impact on the validity of the model but also may overwhelm or cause fatigue to the domain experts. In the *Product Quality* SSI example, for validating the SSI node, the Product Owner decides to specify 10 hypothetical scenarios. One of the designed scenarios is *Code Quality* as “High”, *Testing Status Ratio* as “Medium” and *Software Stability* as “Medium". For this scenario, the Product Owner specifies his perceived most feasible SSI output as "Medium", which coincides with the SSI estimation model output.

6.2.5.2 Outcome Adequacy Validation

This second validation aims at assessing the generalization accuracy of the model using the validation set, i.e., the set withheld from the BN construction steps. For each child node to be validated, the scenarios are extracted from the validation set, instead of being specified by the domain experts like in the Model Walkthrough validation. Two software artifacts *GetCompatibleConfigurations* and *GetChildCompatibleConfigurations* may be used to determine such scenarios automatically from the validation data. For more details on these software artifacts, see Table 8 and the Appendix 1 (section 3). For each one, given that there is no expectation on the child node’s ground truth data availability, the domain experts should provide their judgement or historical record, based on what really happened in such scenario in the historical data. Each of these scenarios is also introduced into the BN, with the aim of comparing the SSI estimation model output to the domain experts’ judgements, recalibrating the corresponding row of the child node CPT under validation when there is a mismatch. Therefore, the main difference of this validation respect to the previous Model Walkthrough validation is the origin of the used scenarios, as the Outcome Adequacy stands on real scenarios instead of hypothetical ones.

Table 8 Tool support for supporting and automating processes of Phase 2

Step	Software artifact name and URL	Input	Output
DAG Specification	**Equal-with binning** (https://github.com/martimanzano/SSI-assessment/blob/TH/src/main/java/Util_Assessment_SI/BayesUtils.java#L213)	-# desired binning intervals - Numerical interval to bin	-Binning intervals with equal width
	Equal-frequency binning (https://github.com/martimanzano/SSI-assessment/blob/TH/src/main/java/Util_Assessment_SI/BayesUtils.java#L213)	-# desired binning intervals - Training set	-Binning intervals with equal frequency
CPTs Specification	**GetFrequencyQuantification** (https://github.com/martimanzano/SSI-assessment/blob/TH/src/main/java/Util_Assessment_SI/BayesUtils.java#L190)	-Training set -Metric's states -Metric's binning intervals	-Quantified frequencies over a metric and its training set
	GetCompatibleConfigurations (https://github.com/martimanzano/SSI-assessment/blob/TH/src/main/java/Util_Assessment_SI/BayesUtils.java#L18)	-Training set -States per metric -Binning intervals per metric	-Set of compatible configurations over the input metrics
	GetChildCompatibleConfigurations (https://github.com/martimanzano/SSI-assessment/blob/TH/src/main/java/Util_Assessment_SI/BayesUtils.java#L45)	-Training set -States per metric -Binning intervals per metric -Child nodes to compute the comp. configs. -(Partial) SSI estimation model	-Set of compatible configurations over the input child nodes
	WSA (https://git.io/Jvspi)	-(Partial) SSI estimation model -Name of the child node under quantification -Parental relative weights toward the child node under quantification	-SSI estimation model with inferred CPT
Estimation Model Validation	**GetCompatibleConfigurations** (https://github.com/martimanzano/SSI-assessment/blob/TH/src/main/java/Util_Assessment_SI/BayesUtils.java#L18)	-Validation set -States per metric -Binning intervals per metric	-Set of observed scenarios over the input metrics
	GetChildCompatibleConfigurations (https://github.com/martimanzano/SSI-assessment/blob/TH/src/main/java/Util_Assessment_SI/BayesUtils.java#L45)	-Validation set -States per metric -Binning intervals per metric -Child nodes to compute the real scenarios. -SSI estimation model	-Set of observed scenarios over the input child nodes

7 SESSI Phase 3: SSI Monitoring

This chapter details the last phase of the SESSI method aimed to put forward the infrastructure for enabling the SSI monitoring. This phase relies on the previous ones for operationalizing the automatic data collection through the data collectors from Phase 1 and their connection to the SSI estimation model built in Phase 2.

The following subsections detail the main elements of the monitoring infrastructure and suggest a generic architecture for putting forward this thesis proposal.

7.1 Main elements of the Monitoring Infrastructure

As mentioned in Chapter 4, the main components for enabling SSI monitoring are:

- The SSI estimation model built in the previous phase. Such models should be deployed and used for estimating SSI and factors. Existing Bayesian Network (BN) APIs tools such as Netica®[20] or unBBayes[21] can be used for such purpose. These software tools include APIs in diverse programming languages like Java or C++ enabling to programmatically use the SSI estimation model. To ease this task, we have implemented a software library named *SSI-assessment* wrapping

[20] https://www.norsys.com/netica.html

[21] http://unbbayes.sourceforge.net/

the unBBayes library and providing a REST API to facilitate the interaction of the SSI estimation models in existing infrastructures. For more details, check the Appendix 1 (section 4).

- The data collectors implemented in Phase 1, in charge of collecting the input data used for feeding the SSI estimation model for monitoring purposes.
- A reporting tool able to show the SSI periodic estimations to the decision makers. Ideally, this tool should also be able to represent the factors and metrics estimations (i.e., have drill-down capabilities) in order to provide interpretability and explainability to such SSI estimations. Examples of reporting tools may be dashboards.

7.2 Monitoring Infrastructure

In order to favour the smoothly integration of the components to enable the SSI monitoring into existing corporate infrastructures, the desired SSI monitoring architecture should be:

- Highly maintainable and testable: new functionalities (as, for example, SSI forecasting and what-if analysis functionalities), as well as changes in the existing components (as, for example, changing the source code of a data collector) should be easy to accommodate in the infrastructure.
- Loosely coupled: all the components of the infrastructure should be weakly coupled to support their high modifiability and to provide an easy integration between the company dashboard and the SSI estimation components.
- Capable of easily integrating visualization/reporting tools such as dashboards to exploit the graphical aspects of the SSI estimation model: this will allow to provide interpretability and explainability to the SSI estimations.
- Capable of easily deploy new versions of SSI estimation models: when a SSI estimation model accuracy decays, the model should be retrained. This retrained model should be able to be easily deployed avoiding major changes.

In line with these characteristics, we suggest the architecture depicted in Figure 17 and detailed below:

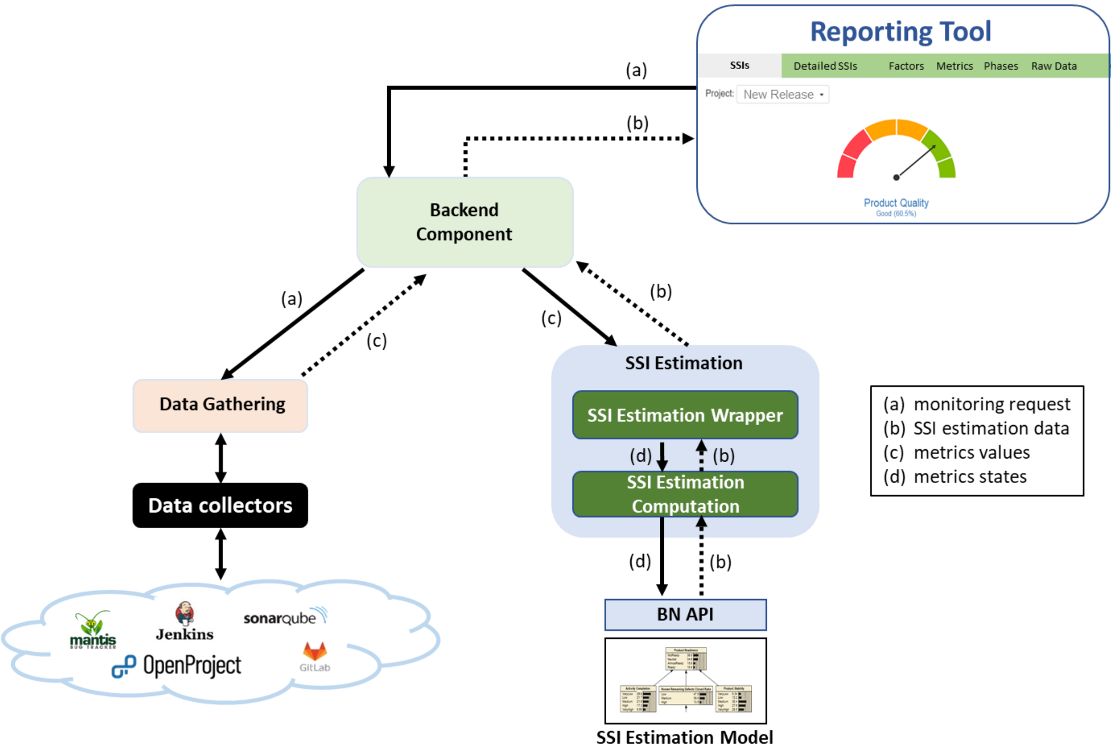


Figure 17 High-level architecture for the SSI monitoring

- The *Reporting Tool* component is in charge of: 1) receiving the monitoring requests from the users and sending them to the *Backend Component* and 2) reporting the data obtained from the *SSI Estimation* component returned by the *Backend Component*.
- The *Backend Component* orchestrates the *Data Gathering* component (to collect the metric values needed to estimate the SSI) and the *SSI Estimation* component (for performing the estimations from the metrics values) to obtain the SSI estimation data.
- The *Data Gathering* component encapsulates the diverse data collectors in order to provide data to other components. To do so, it collects data periodically (metrics values) from the software company's corporate repositories through specific data collectors.
- The *SSI Estimation* component encapsulates the logic to generate the SSI estimations. It receives a set of metrics values and returns the corresponding SSI estimation data (i.e., the probability distributions per node states for the SSI and factors). To do so, it relies on two components:

  - The *SSI Estimation Wrapper* transforms the continuous values of the metrics into the corresponding binned states specified in Phase 2 and feeds these states to the *SSI Estimation Computation* component.
  - The *SSI Estimation Computation* component receives the metrics states and uses the BN API to feed the SSI estimation model to obtain the SSI estimation data.

For more details on this component, refer to the Appendix 1 (section 4).

This proposed architecture has been successfully used in the Q-Rapids project. Chapter 8 provides a specific example of a monitoring infrastructure based on this suggested architecture.

8 Summative Evaluation of the SESSI Method

This chapter details the summative evaluation of the SESSI method led by **RQ3**.

Case study was chosen as the most suitable empirical approach for tackling such summative evaluation. Case studies allow to study in depth the contextual nature of the application of the method. To foster industrial participation, we invited organizations from our industrial collaboration network. The only requirement for them to participate was that they were software companies with an interest in exploiting their corporate repositories to improve their evidence-based decision making. We offered them direct involvement and collaboration of researchers for applying the SESSI method in an action-research fashion (Avison, 2003), i.e., the researchers are directly involved in the application of the method together with the people from the company, as detailed in Chapter 3. The selection of the case study presented here was opportunistic and based on availability and willingness of the company Softeam to participate.

The following sections describe the settings, context, and design of the case study. Afterwards, we detail the execution of the case study. And finally, the feedback, lessons learnt and threats to validity of the case study.

8.1 Case Study Setting

Softeam Group[22] is a large software company located in France, with more than 1650 employees, dedicated to providing business consulting through services and solutions in strategy, consulting, finance, digital, big data, artificial intelligence, analytics, performance, and operations. As a Q-Rapids industrial partner, Softeam promoted the participation of one of their subsidiary firms, namely Modeliosoft[23], which is also a software company providing consultancy services, and additionally developing and maintaining a specific software tool named Modelio.

Modelio is an open-source modelling tool that was first launched in 1991. Currently, Modelio is one of the main modelling software solutions, providing support for standards like UML, BPMN, and ArchiMate, among others. Modelio has a flexible extension mechanism, hence its features can be extended by modules developed by third parties. Modelio's potential users include software developers, business roles, software and system architects, and analysts.

At the time of designing and planning the case study, Modeliosoft was developing an adaptation of Modelio for a specific client. The codename for this software adaptation of Modelio was ModelioNG. They opportunistically selected this product as the software project to which apply the SESSI method in the context of our collaboration, as the product development process was in an initial stage. Therefore, the steps considered in the SESSI method, including the SSI specification, the data collection and the estimation model building would be less disruptive for the ModelioNG development process.

To understand ModelioNG's setting, we analysed some semi-structured interviews conducted with Modeliosoft with the objective of gaining insights on the roles involved, their needs, processes, information flows and project/product management tools used for the product development. Excerpts of such semi-structured interviews are documented and can be consulted in the project's deliverable D3.1 (Q-Rapids, 2019b). Additional insights were gathered informally through face-to-face conversations with

[22] Softeam has recently been acquired by Docaposte. www.softeamgroup.fr

[23] Now Modelio, part of Docaposte. https://www.modelisoft.com

the company representative from the Q-Rapids project during the plenary meetings of such project (from now on, Modeliosoft representative). Based on the compiled information, we summarized the most relevant aspects of the case study settings, which are described as follows.

The software development process of ModelioNG (and in most of the Modelio releases) can be defined as "close to agile" or "customized agile process". It mostly focuses on the production of working software, through close collaboration with customers, an open-source community and face-to-face communication prioritization. Their overall processes and way of working involves market analysis, production of an annual roadmap and the elaboration of strategies to ensure success of their long-lived Modelio product in a competitive environment. When developing specific adaptations of Modelio for customers, Modeliosoft prioritizes the costumers' needs. The planned features and costumers' requests are developed, and major issues are addressed, when any. Regarding Continuous Integration/Continuous Development (CI/CD) adoption and use, they periodically deliver the codebases to integration for nightly build pipelines, including automated tests. Lastly, before delivering new products or versions to their customers, the team ensures the planned features work without blocking issues.

The tools used by the development team when developing new releases/adaptations of Modelio consist of: OpenProject[24] for project management (backlog management, issues, and specification tracking), Mantis[25] for bug tracking and Jenkins[26] for build and test triggering.

Figure 18 illustrates the relation of the roles involved in the case study and the project management tools used during the Modelio development processes.

[24] https://www.openproject.org

[25] https://www.mantisbt.org

[26] https://jenkins.io

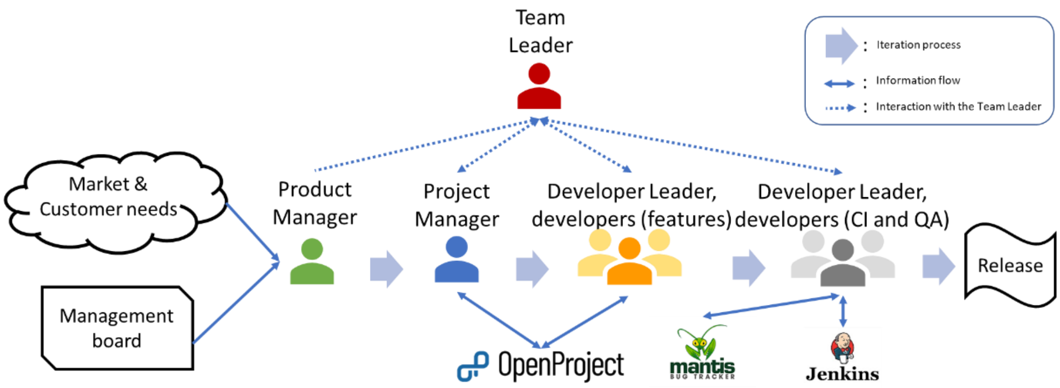


Figure 18 Roles involved in the case study and their relationship with the Modelio development process

8.2 Case Study Design

For the design of the case study, we followed the well-known guidelines provided in Runeson et al. (Runeson et al., 2012). We also consulted Yin (Yin, 2009) for a further understanding of case study research principles. The key research question leading the case study was *"How is the application of the SESSI method in the studied project (ModelioNG)?"*. This research question is linked as an instance of the **RQ3** of the thesis, i.e., "*Is it feasible to apply the SESSI method to specify, estimate, and monitor SSIs for supporting evidence-based decision making in software companies?"* stated in Chapter 1. This case study was aimed at assessing the feasibility of applying the SESSI method in a specific industrial environment provided by Modeliosoft, as part of the third stage of the design science cycle, i.e., the Solution Validation stage.

Specifically, we proposed the participation of 5 researchers as the research team leading the conduction of the SESSI method (including the thesis applicant), from which three of them had previous experience in applying the method in other industrial contexts and were able to provide hands-on support the execution of the method's phases and steps. From the Modeliosoft side, they assigned the team leader of the ModelioNG project as the promotor of the SESSI application in the case study (see Figure 18). The team leader had also the role of Modeliosoft representative. He was our contact point, in charge of providing us access to the required data. He was also involved in the development of the data collectors required for gathering the data to use as input for building the SSI estimation model and to enable the SSI monitoring.

In addition to the mentioned team leader, some other roles participated in the case study as domain experts. Specifically, these participating roles were the product manager, the project manager, the developer leader, and a developer. All of them were involved in the ModelioNG project. To assign the participating roles for each phase and step of the case study, the team leader decided and assigned the roles, according to their expertise and availability. Our interaction with these domain experts was direct or indirect, depending on their availability.

The case study design was flexible to deal with daily unexpected issues. Details of issues and/or decisions taking during the case study are detailed in the execution section.

8.3 Data Collection and Data Analysis

Data collection and data analysis were performed according to the phases and steps of the SESSI method described in Chapter 5 and 6. For instance, after studying the ModelioNG context and supporting the specification of the SSI, data collectors were developed for gathering data. This data together with domain experts' knowledge was used as the basis for constructing the estimation model. These activities were based on the guidelines of the SESSI method as it is further explained in the next subsections. We also used individual diaries to record all interactions, issues and relevant observations from the execution of the method. Further details of collected information cannot be provided given non-disclosure agreements with Modeliosoft.

In addition to the SESSI method procedures, we designed a survey based on a questionnaire as a data collection instrument for gathering practitioners' feedback. We used a previously defined questionnaire from the Q-Rapids project (Q-Rapids, 2017) as the basis for designing this one. The questionnaire was also piloted and approved by the team leader. As the questionnaire contained mostly closed questions, we processed the gathered data using spreadsheets. For the case of open questions, we planned to use content analysis for analyzing and categorizing all the responses (Krippendorff, 1980). The questionnaire was designed with the aim of being simple and brief, so it could be filled in 10-15 minutes.

8.4 Case Study Execution – SESSI Phase 1: SSI Specification

After internal discussions in Modeliosoft, the product manager of the ModelioNG project chose *Product Readiness* as the SSI to be tackled by the SESSI method, as it was communicated to us by the team leader.

One the one hand, regarding the SSI textual definition, we held some discussions mainly by email with the Modeliosoft representative and the product manager, as they were the main roles taking decisions related to the *Product Readiness* SSI. They specified the SSI textual definition. They included the overall rationale of the SSI and a summary of the high-level basis required to assess it. The textual definition was stated as "*Product Readiness provides high level information on product readiness for the next release. A product ready to be released implements the features planned for the release and without critical bugs*".

One the other hand, regarding the SSI hierarchical decomposition, we agreed on first conducting research on the selected *Product Readiness* SSI in the scientific literature, in order to provide them with evidence and examples of how the SSI is computed in other contexts. These examples were meant to serve as a starting point to ease the SSI specification process. We found some works providing definitions and estimation techniques for the product readiness (Asthana and Olivieri, 2009; Staron et al., 2012), and shared them with Modeliosoft through the team leader, as he was our contact point. However, none of them fitted their specific understanding and needs for the SSI in the ModelioNG project, i.e., their product rules, including the criteria to categorize their products under development as ready to be released or delivered, as these criteria were different to the ones used in those research works.

In contrast, the product manager and the team leader compiled the list of aspects relevant for their readiness criteria, based on their product and business rules. They sent such list to us, and after some iterations, discussions, and refinements, we derived the final *Product Readiness* SSI specification. An example of a refinement to the original specification delivered by Modeliosoft members includes the inclusion of an additional factor and corresponding metric reflecting the postponing of low-severity issues.

The resulting *Product Readiness* SSI specification is shown in Figure 19. It is based on 3 factors and 6 metrics directly computed from Modeliosoft's corporate repositories.

We did not have access to the disaggregated raw measures due to confidentiality reasons. More details on the rationale of the factors and metrics are presented in Table 9.

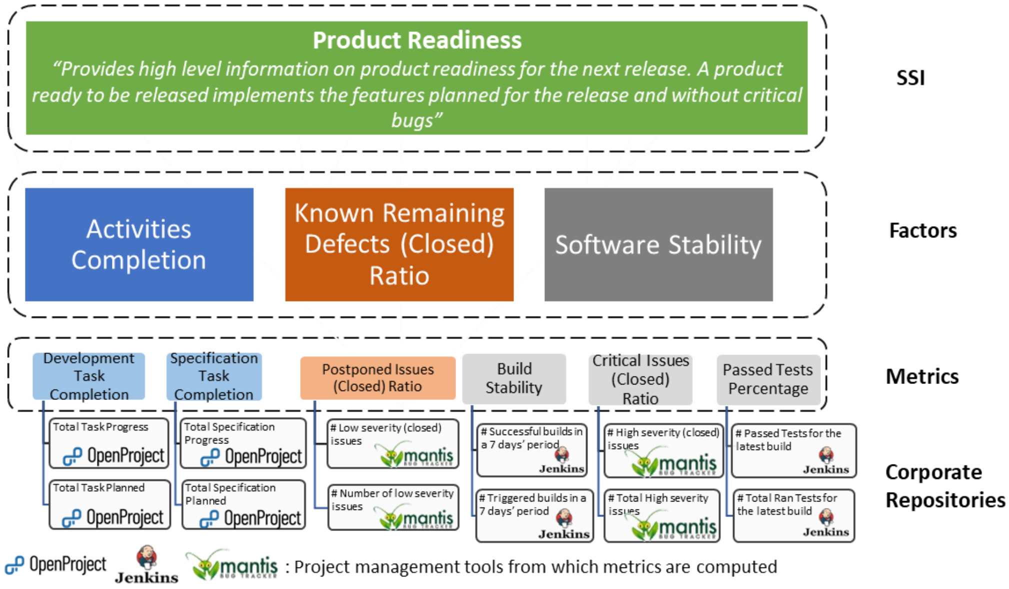

Product Readiness
"Provides high level information on product readiness for the next release. A product ready to be released implements the features planned for the release and without critical bugs"
SSI
Activities Completion
Known Remaining Defects (Closed) Ratio
Software Stability
Factors
Development Task Completion
Specification Task Completion
Postponed Issues (Closed) Ratio
Build Stability
Critical Issues (Closed) Ratio
Passed Tests Percentage
Metrics
Total Task Progress
OpenProject
Total Task Planned
OpenProject
Total Specification Progress
OpenProject
Total Specification Planned
OpenProject
# Low severity (closed) issues
mantis
# Number of low severity issues
mantis
# Successful builds in a 7 days' period
Jenkins
# Triggered builds in a 7 days' period
Jenkins
# High severity (closed) issues
mantis
# Total High severity issues
mantis
# Passed Tests for the latest build
Jenkins
# Total Ran Tests for the latest build
Jenkins
Corporate Repositories
OpenProject
Jenkins
mantis
: Project management tools from which metrics are computed

Figure 19 Specification of *Product Readiness* SSI in the ModelioNG case study

Table 9 Specification of the *Product Readiness* SSI for the ModelioNG case study

Textual Definition: *Product Readiness provides high level information on product readiness for the next release. A product ready to be released implements the features planned for the release and without critical bugs.*					
Factor	**Description**	**Metric**	**Description**	**Data Source**	**Definition** (In some cases, the metric's definition shown is simplified)
Activities Completion	Represents the status of the completion of the activities plan for this release, including development and specification tasks	Specification Task Completion	Represents the fulfilment of the required specification tasks for this release	Open Project	$\frac{total\_specification\_progress}{total\_specification\_planned}$ Where: $total\_specification\_progress = \sum_{i=1}^{N\ WPs\ of\ type\ "Specif."} WP\_i.spent\ time$ $total\_specification\_planned =$ $= \sum_{i=1}^{N\ WPs\ of\ type\ "Specifi."} WP_i.spent\ time$ $+ WP_i.remainingtime$
		Development Task Completion	Represents the fulfilment status of the required development tasks for this release	Open Project	$\frac{total\_task\_progress}{total\_task\_planned}$ Where: $total\_task\_progress = \sum_{i=1}^{N\ WPs\ of\ type\ "Task"} WP\_i.spent\ time$ $total\_task\_planned =$ $= \sum_{i=1}^{N\ WPs\ of\ type\ "Task"} WP\_i.spent\ time$ $+ WP\_i.remainingtime$

Known Remaining Defects (Closed) Ratio	Measures the defects/ bugs/crashes that lie outside the major bug category and can be deferred to next releases	Postponed Issues (Closed) Ratio	Ratio of the minor severity closed issues (of type Feature/ Trivial/ Text/ Tweak/ Minor/ Usability) with respect to the total number of low severity issues	Mantis	$\frac{no.of\ low\ severity\ closed\ issues}{no.of\ low\ severity\ closed\ issues + no.of\ low\ severity\ open\ issues}$
Software Stability	Measures the status of the operational software quality of the monitored release, taking into consideration the presence of major issues and the testing status	Build Stability	Percentage of successful builds with respect to the total of builds triggered in a seven days' period	Jenkins	$\frac{successful\ builds}{total\ builds\ triggered}$
		Critical Issues' (Closed) Ratio	Ratio of high severity closed issues (of type Crash/ Block/ Major) with respect to the total number of high severity issues	Mantis	$\frac{no.of\ high\ severity\ closed\ issues}{no.of\ high\ severity\ closed\ issues + no.of\ high\ severity\ open\ issues}$
		Passed Tests Percentage	Percentage of tests passed with respect to the total number of tests ran for the latest build	Jenkins	$\frac{tests\ passed}{total\ tests\ ran}$

Regarding the data collectors' availability, Modeliosoft had a data collector for Jenkins. However, they did not have data collectors for the rest of the project management tools required for the SSI estimation. Therefore, after finishing the specification of the *Product Readiness* SSI, some data collectors had to be developed for the project management tools that had not any at that point, in order to automatically extract, compute and store the metrics specified in Table 9.

Modeliosoft developed data collectors for Mantis and OpenProject. For the case of Jenkins, they reused data collectors[27] developed in the context of the Q-Rapids project. Once their development was finished, the data collectors were configured to run daily for collecting data and computing and storing the metrics values. The computed values were dumped into an Elasticsearch[28] document-based database with the same daily frequency. Apart from collecting historical data to be used later on in the second SESSI phase, these data collectors allow the SSI monitoring once its estimation model has been created, as the daily collected data can be fed into the SSI estimation model. The data

[27] https://git.io/JvGwf

[28] https://elastic.co

collectors are open-source, and their code is freely available on the Q-Rapids GitHub repository[29].

8.5 Case Study Execution – SESSI Phase 2: SSI Estimation

For the purpose of creating such SSI estimation model, we designed a set of guidelines and tool support previously described in Chapter 6 and in the Appendix 1, with the aim of supporting and easing the steps in the case study. The main aspect we tried to ease was the elicitation of probabilities, as the Modeliosoft participants were not familiar with determining large sets of probabilities. Additionally, they had tight schedules and time constraints so we had to adapt the case study tasks to these restrictions.

Before starting the steps of this phase, we first needed to collect historical data to use during the steps of this phase. Such data collection process was performed through data collectors automatically gathering and storing the metrics values of the *Product Readiness* SSI. Such data collectors ran and collected such data with a daily frequency.

Due to their time availability, and to ensure having enough data to be able to conduct the second phase of the SESSI method smoothly, the ModelioNG team, together with our research team, decided to perform this second phase of the method to build the estimation model for the *Product Readiness* SSI once we had a period of 3 months of collected data. Five different roles part of the ModelioNG development team participated in this phase: the product manager, the project manager, the developer leader, the team leader, and a developer.

In the following subsections, we describe how the steps of the SSI estimation model building were conducted in Modeliosoft for the ModelioNG case study.

8.5.1 Data Splitting

The research team oversaw this step and had access to the ModelioNG's historical data through anonymized data snapshots periodically sent by the Modeliosoft representative.

For conducting this step, we, as research team, configured and deployed a local, Elasticsearch document-based database instance, and restored the data snapshots periodically sent by the Modeliosoft representative.

[29] https://github.com/q-rapids

The historical data contained in the data snapshots was composed of the *Product Readiness* metrics historical values, yielding in the [0,1] continuous interval. They were daily collected by the data collectors deployed in their company premises during the first phase of the SESSI method.

In this first step, we divided the entire period of collected historical data into two splits: the training set, to be used for the model building, and the validation set, to be withheld from the model building steps for its use in the validation step. We performed the data splitting using 80% of the historical data for the training set, and the remaining 20% was reserved as validation set.

8.5.2 DAG Specification

This step consisted in the specification of the graphical structure of the Bayesian Network (BN) estimation model for the *Product Readiness* SSI, i.e., the BN's Directed Acyclic Graph (DAG). The DAG, as previously explained, is composed by nodes representing the hierarchical components of the SSI specified in the previous phase of the SESSI method. Directed edges are inserted when causal relationships exist between two nodes.

To perform this step, we held two virtual meetings with the team leader in order to complete a preliminary DAG. The specified DAG was composed of the metric nodes yielding at the bottom level of the hierarchy and impacting the factor nodes at the mid-level of the hierarchy, which at their turn impact the SSI at the top level of the hierarchy. No relations (i.e., edges) were placed among nodes yielding on the same level. The specified DAG is shown in Figure 20.

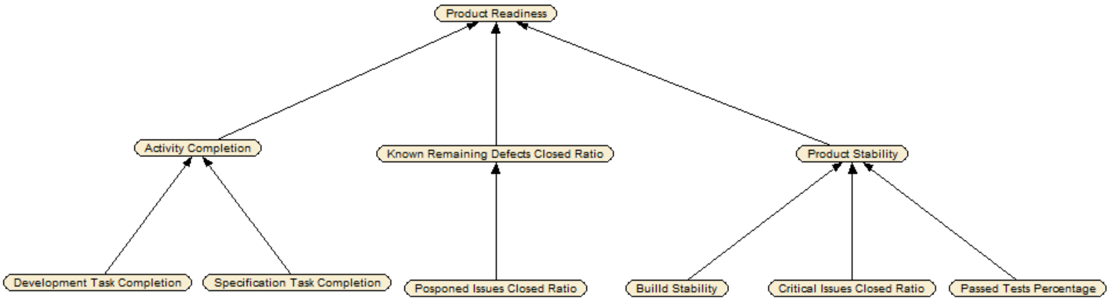

Product Readiness
Activity Completion
Known Remaining Defects Closed Ratio
Product Stability
Development Task Completion
Specification Task Completion
Posponed Issues Closed Ratio
Builld Stability
Critical Issues Closed Ratio
Passed Tests Percentage

Figure 20 DAG specified for the *Product Readiness* SSI estimation model

8.5.3 CPTs Specification

After the DAG specification, we prepared detailed instructions in order to ease the remaining tasks of this step: the specification of the states for the BN nodes and the binning intervals for the metric nodes, whose values were being collected in the continuous interval [0, 1] by the data collectors.

We sent the detailed instructions to the team leader, who defined the states for each node, according to the product rules commonly used in the Modelio projects. He specified ordinal states for all the nodes, although the set of specified states was not equal for all the nodes. Table 10 shows the specified states for each node composing the *Product Readiness* SSI estimation model.

Table 10 Specified states for the nodes composing the *Product Readiness* SSI estimation model

Node	**Type**	**States**
Development Task Completion	Metric	Very Low, Low, Medium, High, Very High
Specification Task Completion	Metric	
Critical Issues (Closed) Ratio	Metric	
Build Stability	Metric	
Activity Completion	Factor	
Product Stability	Factor	
Passed Tests Percentage	Metric	
Known Remaining Defects (Closed) Ratio	Factor	Low, Medium, High
Postponed Issues (Closed) Ratio	Metric	
Product Readiness	SSI	Not Ready, Neutral, Almost Ready, Ready

To ease the specification of the binning intervals, we used our implemented versions of the two main unsupervised binning algorithms described in Chapter 6 (Equal-Width and Equal-Frequency binning). We computed binning intervals for each metric node using both binning algorithms, and then showed the resulting binning intervals to the team leader to use them as a starting point. He defined his own binning intervals for each metric node according to their implicit rules and metrics’ thresholds. We show an excerpt of the specified binning functions in Table 11 for the *Build Stability* node. The rest of the binning intervals specified for the remaining metric nodes can be found in the Appendix 2 (section 1, Table A1).

Table 11 Specified binning intervals and quantified frequencies for the *Build Stability* node

Build Stability		
State	**Interval**	**Quantified frequency**
Very Low	[0, 0.4)	3%
Low	[0.4, 0.7)	4%
Medium	[0.7, 0.8)	8%
High	[0.8, 0.95)	25%
Very High	[0.95, 1]	60%

To specify the Conditional Probability Tables (CPTs) for each node, we used supporting tools enabling the semi-automatically filling of the CPTs (see Chapter 6 and Appendix 1). We used the training set resulting from the data splitting step, previously explained in section 8.5.1. Three domain experts from the ModelioNG team participating in the case study were involved in this step: the team leader, the project manager, and the developer leader.

The CPT specification process was performed starting from the metric nodes, then the factor nodes, and lastly the SSI node. This enabled the usage of the software artifacts to automatically compute the Weighted Sum Algorithm (WSA) required compatible configurations for the case of the SSI. We detail the process according to the type of node and the order in which the CPTs were filled, as follows:

1. **Metric nodes:** We used frequency quantification over the training set and domain experts' knowledge to fill in the CPTs for these nodes. For this purpose, we used our developed software artifact *getFrequencyQuantification* over the training set, the states defined for each of these nodes, and their corresponding binning intervals. The domain experts reviewed and refined the automatically computed CPTs, according to their knowledge and product rules. In Table 11 (third column) we show the resulting probabilities per category for the *Build*

Stability node. This node, whose values were collected from Jenkins as the percentage of successful builds in a 7 days period in the [0,1] continuous interval, had 5 ordinal states defined by the domain experts, ranging from “Very Low” to “Very High”, and corresponding binning intervals defined in the previous step. The automatically computed CPT for this metric through frequency quantification ranged from 3% for “Very Low”, to 60% for “Very High”, as software builds succeeded most of the time in the training set data. The rest of the metric nodes CPTs were also computed through frequency quantification and refined by the domain experts. Such CPTs are shown in the Appendix 2 (section 1, Table A1).

2. **Factor nodes:** CPTs for these nodes varied in size. The CPT for the factor node *Known Remaining Defects (Closed) ratio* only had a parent node (*Postponed Issues (Closed) Ratio*) with only 3 categories (“Low”, “Medium”, “High”), so it resulted in 9 probabilities in total for its CPT (3 rows of 3 probabilities each). This CPT was manually filled in by the domain experts adding uncertainty in the relation between the factor node and its corresponding unique metric. The CPT of this node can be found in the Appendix 2 (section 1, Table A2). The rest of CPTs for these nodes were large, as these nodes had more than a parent node, and hence would have implied an excessive number of entries to be filled in by the domain experts. Therefore, we applied the WSA technique in order to reduce the number of probabilities to elicit. We computed the compatible configurations required by the WSA automatically from the training data, using our tool *getCompatibleConfigurations* previously explained. With the compatible configurations computed, we prepared another supporting asset to ease the child nodes’ probabilities elicitation. The domain experts were requested to provide the resulting probabilities for each compatible configuration, and, additionally, the relative weights of the parent nodes towards the child node under quantification. Using this information, our implementation of the WSA inferred the remaining probabilities of the CPTs. Without the WSA, the CPT of the factor node *Activity Completion* would have required filling in 125 probabilities (25 rows of 5 probabilities each). The *Product Stability* node would require 625 probabilities (125 rows of 5 probabilities each) for its CPT, apart from manually specifying the compatible configurations. In contrast, by

applying our implementation of the WSA, the number of probabilities to be provided by domain experts decreased to 10 rows of 5 probabilities each for the *Activities Completion* node, and 15 rows of 5 probabilities each for the *Product Stability* node. Furthermore, the compatible configurations were automatically extracted from the training set data, thus not needing to be specified by the domain experts. The domain experts only had to provide the resulting probabilities for the node under quantification for each compatible configuration, along with the parent nodes relative weights required by the WSA. An excerpt of such compatible configurations computed with the *getCompatibleConfigurations* tool, along with the provided resulting probabilities and relative weights elicited from domain experts is shown in Table 12 for the *Product Stability* node. The partial CPT elicited for the *Product Stability* and *Activity Completion* nodes are shown in the Appendix 2 (section 1, Table A3, Table A4, respectively).

Table 12 Excerpt of the compatible configurations for the factor node *Product Stability*, along with the elicited probabilities and relative weights required by the WSA

Parent nodes			**Product Stability**				
Build Stability (W=20%)	**Critical Issues (Closed) Ratio (W=40%)**	**Passed Tests Percentage (W=40%)**	**Very Low (%)**	**Low (%)**	**Medium (%)**	**High (%)**	**Very High (%)**
Very Low	Very Low	Very Low	90	10	0	0	0
Very Low	Low	Very Low	70	20	10	0	0
Low	Low	Low	55	40	5	0	0
Medium	High	Medium	3	25	50	20	2
High	Medium	High	3	25	50	20	2
High	Very High	Very High	0	3	7	20	70
Very High	Very Low	Very High	8	10	62	15	5

3. **SSI node:** The case of the SSI node was similar to the case of factor nodes with large CPTs. However, in this case, in order to automatically compute the compatible configurations for the WSA using the *getCompatibleConfigurations*

artifact, we needed a partially constructed BN, as there was no historical data collected for the factor nodes impacting such SSI node. Therefore, we built a partial BN using the specified DAG and the CPTs for the rest of the nodes. Afterwards, we computed the compatible configurations for the factor nodes impacting the SSI node using the *getChildCompatibleConfigurations* software artifact. Finally, we elicited the resulting SSI probabilities for each configuration, and the relative weights of the parent factor nodes towards the SSI node. Manually filling the *Product Readiness* CPT would have required eliciting 300 probabilities in total (75 rows of 4 probabilities each). In contrast, by using the WSA, we reduced such number to 13 rows of 4 probabilities each, apart from not requiring the manual specification of the compatible configurations. The partial CPT elicited for the SSI node is provided in the Appendix 2 (section 1, Table A5).

8.5.4 Estimation Model Generation

For the SSI estimation model construction (either for the partially constructed BN required by the *getChildCompatibleConfigurations* software artifact and the WSA), we used the Netica® software through its graphical user interface. Each BN node was represented as a discrete node with its specified states, and edges representing conditional relationships were added between them as specified in the DAG Specification step (section 8.5.2). For clarification purposes, we added annotations showing the specified binning intervals and the WSA relative weights, for the case of parent nodes whose child nodes' CPTs were defined using the WSA technique.

This step was performed by the research team, as it involved the usage of the Netica® software, with which the ModelioNG team did not have experience with. The outcome of this step consisted in a BN estimation model ready to be validated in the next step of the SESSI method. Such resulting BN is shown in Figure 21, including the DAG structure and the nodes prior probabilities. Such prior probabilities show how probable is for each node to be in each of its states before entering any evidence into the model. These prior probabilities are derived from the nodes' CPT.

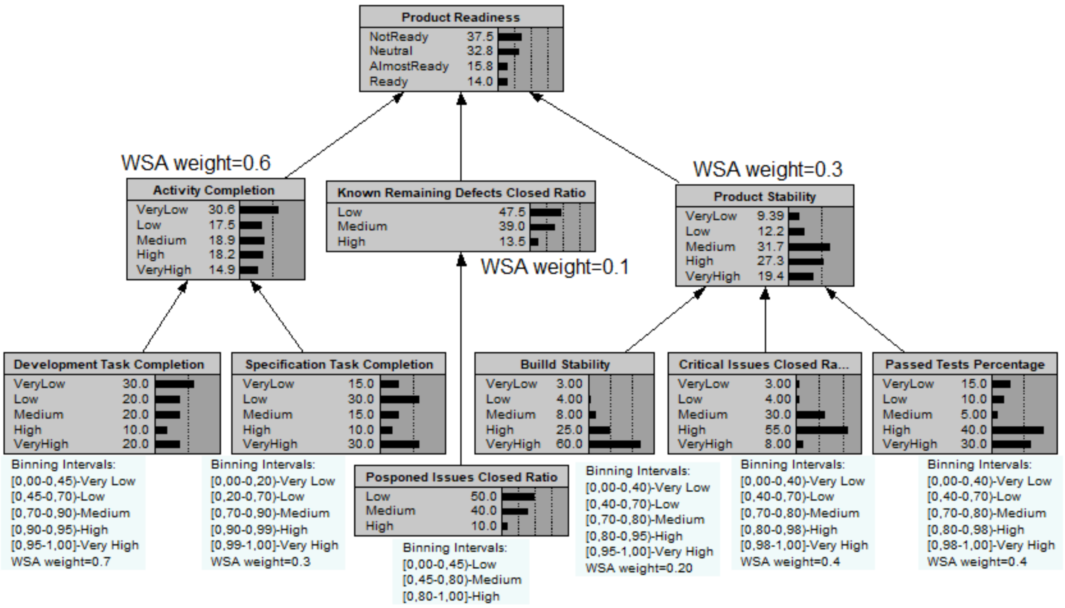


Figure 21 *Product Readiness* BN estimation model resulting from the 3 first steps

8.5.5 Estimation Model Validation

Once the initial BN estimation model for the *Product Readiness* SSI was generated, we conducted the two validations considered in the SESSI method and described in Chapter 6: Model Walkthrough and Outcome Adequacy. Following the SESSI guidelines presented in the previous chapters, we performed the validations over the child nodes. The two validations were conducted sequentially, according to their degree of evidence. In both cases we prepared supporting assets to ease the tasks to be fulfilled by the domain experts.

We describe the conducted process and the outcome of both validations in the following subsections.

8.5.5.1 Model Walkthrough Validation

This validation aimed to test and recalibrate the model using hypothetical scenarios and the domain experts' perceptions. Three ModelioNG members participating in the case study were involved in this step: the project manager, the team leader, and a developer. We prepared and delivered them supporting material to ease the task of specifying the hypothetical scenarios and their expected resulting state with highest probability for each node to validate.

A total of 41 hypothetical scenarios and their expected resulting states were provided by the domain experts. Afterwards, we introduced each scenario in the BN estimation model as a "what-if" scenario and compared the child node state with highest accuracy yielded by the BN estimation model to the state perceived by the domain experts. In cases where there was a mismatch, we recalibrated the corresponding child node CPT row by tuning the resulting probabilities, to make them coincide with their perception. We also revised and tuned, when required, the CPT rows above and below the recalibrated one. Table 13 shows a summary of the Model Walkthrough validation performed for each node, showing, for each validated node, the number of scenarios designed by the experts, the number of mismatches that required model recalibration and the percentage of matches or accuracy.

For instance, for the *Product Stability* node, the domain experts were asked to provide 14 hypothetical scenarios. As an example, one of the designed scenarios was *Build Stability* as "Medium", *Critical Issues (Closed) Ratio* as "Very High" and *Passed Tests Percentage* as "Medium". For this scenario, the experts specified the most probable state for the *Product Stability* node as "Medium", which matched the output of the estimation model. Individual tables showing the conducted Model Walkthrough are shown in the Appendix 2 (section 1, Table A6, Table A7, Table A8, and Table A9). The average accuracy obtained in this validation was suboptimal due to the high number of scenarios that required recalibration for the *Activity Completion* node, probably due to the similarity among the scenarios used as compatible configurations for the WSA.

Table 13 Summary of the Model Walkthrough validation conducted in the ModelioNG case study for the *Product Readiness* SSI

Node	**Number of scenarios designed**	**#Required recalibration**	**Matches (%)**
Activity Completion	12	7	41,6
Product Stability	14	4	71,4
Known Remaining Defects (Closed) Ratio	1	0	100
Product Readiness	14	4	71,4
Total	**41**	**15**	**63,4**

Figure 22 shows the recalibrated estimation model resulting from the Model Walkthrough Validation. It can be noted how the nodes probabilities slightly changed for the recalibrated nodes (mainly the *Activities Completion* node as it was the node which required most recalibrations), with respect to the estimation model shown in Figure 21 (before conducting this validation).

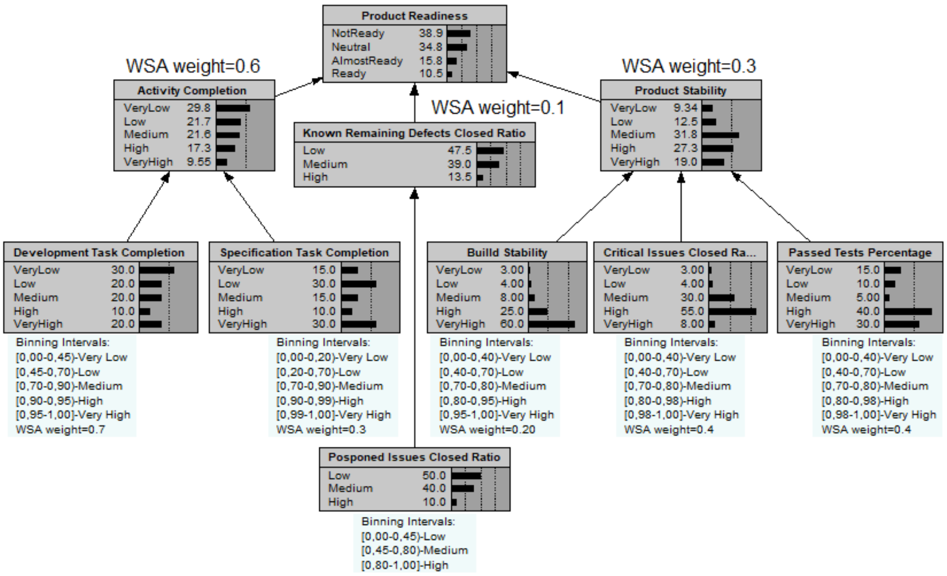


Figure 22 *Product Readiness* SSI estimation model after conducting the Model Walkthrough validation

8.5.5.2 Outcome Adequacy Validation

This validation was carried out right after finishing the Model Walkthrough validation, using the partially validated SSI estimation model resulting from the first validation. We extracted the real scenarios using the validation set of the historical data. For the factor nodes, such real scenarios (i.e., combinations of states of the metric nodes) were directly extracted from the validation set using the software artifact *getCompatibleConfigurations*, as the parent nodes values were directly observable (as they corresponded to the discretized metrics continuous values). For the SSI node, as its parent nodes were not directly observable in the validation set (as they were defined during the case study and yielded in the SSI estimation model), we used the *getChildCompatibleConfigurations* artifact along with the preliminary SSI estimation model, which allowed us to extract the real, observed scenarios from the validation set.

We prepared and delivered detailed instructions to the domain experts to ease the elicitation of the resulting states for the nodes to validate, according to what happened in each observed scenario. For each one, the domain experts were requested to specify the resulting state for the node under validation according to what happened in the observed scenario. Their answer was then compared to the estimation model output, after entering the scenario into the model such as in the Model Walkthrough validation. For the cases in which there were mismatches, the CPT corresponding to the node under validation was modified by tuning the probabilities of the corresponding row. Then, we revised and tuned, when required, the CPT rows above and below the recalibrated one, as performed in the Model Walkthrough validation.

Table 14 shows a summary of the results of this validation for each validated node. For each one, Table 14 shows the number of real scenarios from the validation set, the number of mismatches that required recalibration, and the percentage of matches between the domain experts' answers and the model output (accuracy). For instance, for the *Product Stability node,* there were 10 real scenarios in the validation set, and domain experts provided the resulting state for each of these scenarios. 3 out of these 10 scenarios resulted in mismatches between the states specified by the domain experts and the states yielded by the SSI estimation model. Individual tables showing the conducted Outcome Validation are shown in the Appendix 2 (section 1, Table A10, Table A11, Table A12, and Table A13).

Compared to the Model Walkthrough validation, there was a smaller number of recalibrations, thus yielding a higher overall accuracy. After recalibrating the model in the required cases, we obtained the final estimation model for the *Product Readiness* SSI. Such final model is shown in Figure 23. It can be seen how the nodes probabilities slightly changed for the validated and recalibrated nodes with respect to the one of Figure 22 (after the Model Walkthrough validation).

Table 14 Summary of the Outcome Adequacy validation conducted in the ModelioNG case study for the *Product Readiness* SSI

Node	**Number of real scenarios considered**	**Required recalibration**	**Matches (%)**
Activity Completion	1	0	100
Known Remaining Defects (Closed) Ratio	1	0	100
Product Stability	10	3	70
Product Readiness	6	1	83,3
Total	**18**	**4**	**77,7**

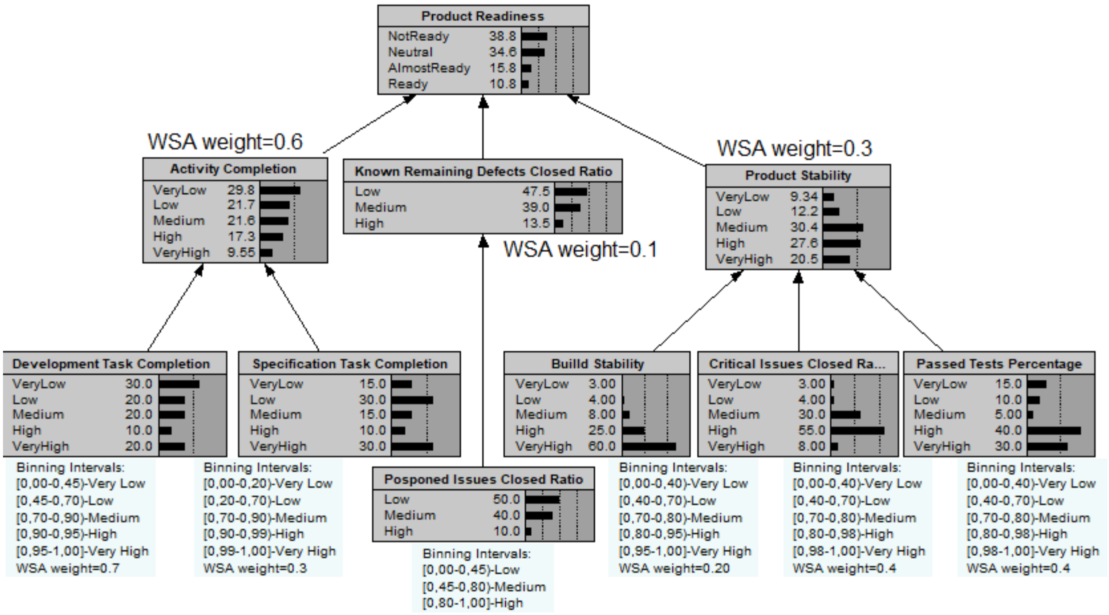


Figure 23 *Product Readiness* estimation model after conducting the two validation steps

8.6 Case Study Execution – SESSI Phase 3: SSI Monitoring

After obtaining the final SSI estimation model for the *Product Readiness* SSI, we delivered it to Modeliosoft. We also discussed with them the most feasible alternatives for its deployment in the company.

Since the very beginning of the case study, Modeliosoft stated their interest on integrating the resulting estimation model into a dashboard that was being promoted by their headquarter Softeam to visualize and monitor SSIs in the context of the Q-Rapids project. Therefore, to enable the integration of the SESSI infrastructure into such dashboard, we adapted the architecture shown in Chapter 7. Such resulting architecture is shown in Figure 24 and detailed as follows.

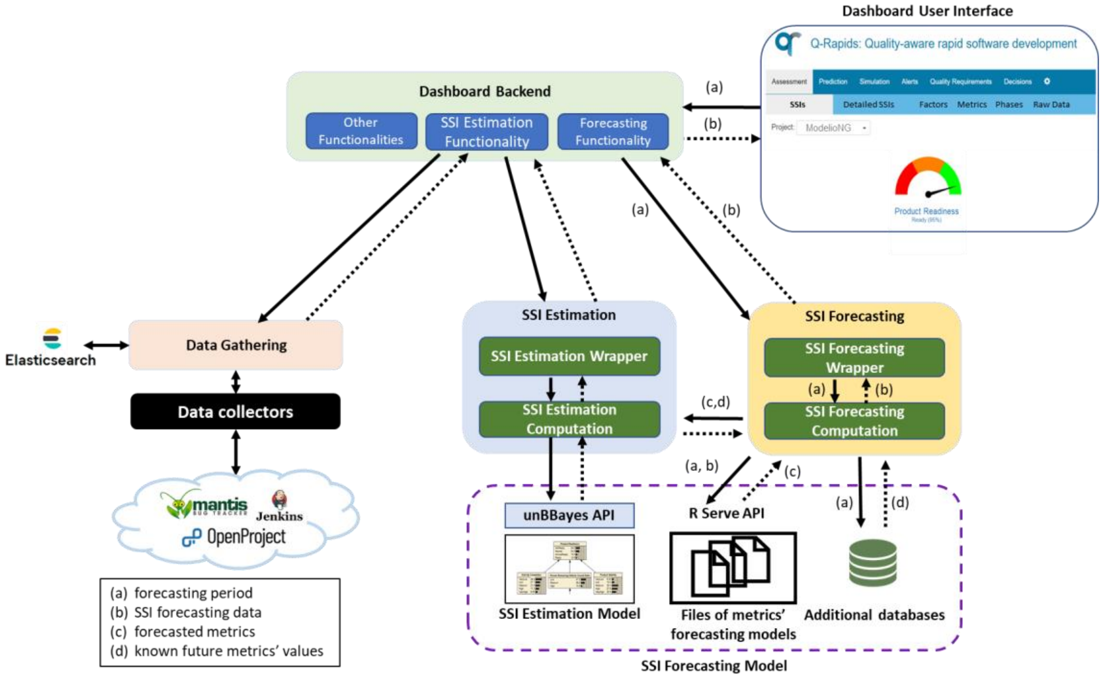


Figure 24 High-level architecture for the SSI monitoring through a dashboard and data collectors in Modeliosoft

- The *Dashboard User Interface*[30] is the frontend of the dashboard instantiating the *Reporting Tool* suggested in the generic infrastructure from Chapter 7.
- The *Dashboard backend*[31] is the backend of such dashboard, in charge of: 1) receiving the requests from the *Dashboard User Interface* and sending them to the *SSI Estimation* component through the *SSI Estimation Functionality* and 2) providing the *Dashboard User Interface* with the data obtained from such *SSI Estimation* component. The formatting of the data to be displayed in the frontend is performed by the *SSI Estimation Functionality* component.

[30] Q-Rapids-dashboard (User Interface) (https://git.io/JvG0Z)

[31] Q-Rapids-dashboard (Backend) (https://git.io/JvG0Z)

- The *SSI Estimation Functionality*[32] component orchestrates the *Data Gathering* component (to collect the metrics values of the SSI) and the *SSI Estimation* component to obtain the SSI estimation data.
- The *Data Gathering*[33] component is an instantiation of the generic component reported in Chapter 7 with connectors for the project management tools required to gather the metrics for estimating the *Product Readiness* SSI, i.e., OpenProject, Mantis, and Jenkins.
- The *SSI Estimation*[34] component encapsulates the logic to generate the estimations. This component is the same as the suggested in the generic infrastructure in Chapter 7. More details on this component may be found in such chapter and in the Appendix 1 (section 4).

Additionally, in Modeliosoft, they were also interested in being able to conduct "what-if" analysis to assess scenarios that could help them to take preventive actions, with the aim of reducing the risk of delivering the software product without meeting their product requirements and identifying opportunities. To enable such analysis, we used Netica® software, as it allows to interact with the SSI estimation models easily and through a graphical interface. Figure 25 shows an example of a "what-if" analysis conducted with the *Product Readiness* SSI estimation model, with a manually entered scenario in which the development of features is almost finished (that is, the *Activities Completion* related nodes, i.e., *Development Task Completion* and *Specification Task Completion* in their "High" and "VeryHigh" states, respectively). The percentage of minor bugs addressed is low, as well as every *Product Stability* parent node (*Build Stability, Critical Issues (Closed) Ratio*, and *Passed Tests Percentage* nodes*).* Given this scenario, the SSI estimation model propagates the probabilities to the remaining, unobserved nodes, yielding the "Not Ready" SSI state as the most probable. This is because even when the features to deliver are almost completed, the stability of the software and the percentage of non-closed minor bugs are deficient and thus not meeting Modeliosoft's readiness requirements.

[32] Q-Rapids-dashboard (https://git.io/JvG0Z)

[33] Q-Rapids connect (https://git.io/JvGwf)

[34] SSI-assessment (https://github.com/martimanzano/SSI-assessment/)

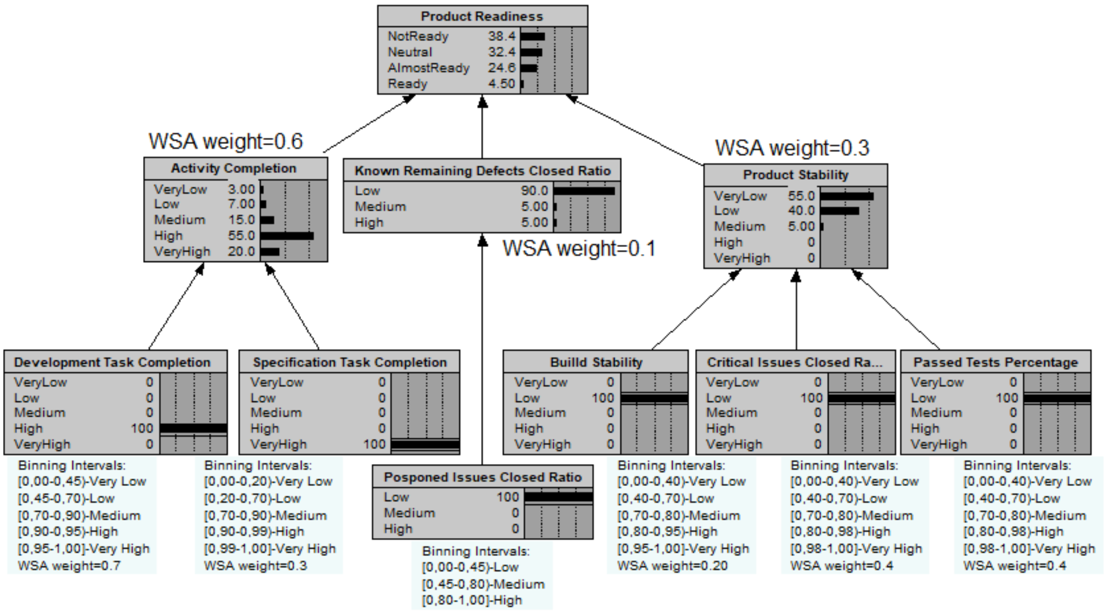


Figure 25 "What-if" analysis example using the *Product Readiness* SSI estimation model resulting from the case study

These "what-if" analysis can be performed graphically using software such as the mentioned Netica® or unBBayes (which has also a graphical interface allowing the interaction with BNs), or programmatically using the *SSI estimation* component.

8.7 Feedback from the case study

In addition to the knowledge obtained by actively participating in the case study and interacting with the participants from Modeliosoft, we were especially interested in inquiring about practitioners' perceptions regarding the method execution and its potential usefulness. With this aim, right after the execution of the SESSI method, we requested feedback on the application of the method to the case study participants.

The feedback gathering procedure was performed as follows: we designed a questionnaire as a data collection instrument. This questionnaire can be consulted in the Appendix 2 (section 2). It was designed with the aim of being simple and brief, so it could be filled in 10-15 minutes at the end of the case study execution. The questionnaire contained open and closed questions organized into three main sections. Each one of these sections focused on:

1. Ranking the execution of the SESSI method as: usable, clear, difficult, reliable, complete, comprehensive, and repeatable.

2. Positive/negative aspects of the method observed by the case study participants during its execution.
3. Opinion on the reproducibility of the method in another case or context without our support.

This questionnaire was emailed to the Modeliosoft representative right after the execution of the SESSI Phase 3 in the case study. Although we requested each participant to fill in the questionnaire, the representative provided us with a single set of answers that was collaboratively agreed among all participants during an internal meeting. We did not have knowledge nor control on this meeting and the resulting answers. We rely on these answers as a representative agreement among all the participants.

The feedback was positive for most of the closed requested aspects. In particular, the aspects of completion, reliability, comprehensibility, detail, interest, and repeatability got the highest scores. None of the closed requested aspects was negatively scored.

The self-explanatory aspect of the SESSI method was scored as neutral, which is related to one of the positive open aspects of the method execution as emphasized by the company's participants: the "understandability of the process and clear explanations of the different steps by the research team". On the other hand, as negative aspects, they highlighted that "historical data selection [for the phase 2 of the method] was not totally clear". We realize that interpretability aspects such as the historical data selection are critical when providing methods, mechanisms, or tools to managerial roles, as they do not necessarily understand technical details and prefer to have a high-level view.

Finally, with respect to the repeatability of the method, participants agreed that after this execution their general perception was that they would be able to repeat the method by themselves without our help.

Our assessment from the received feedback led us to confirm most of the informal feedback and comments we received during our involvement with Modeliosoft participants in the ModelioNG *Product Readiness* case study: On one hand, while the positive perceptions regarding the execution of the method might be biased by our own implication as participants in the case study, the participants from the company never faced directly any problem or challenge regarding the execution of the method, as we were the ones in charge of fitting and guiding their activities during the case study. At this respect, although we had experience in the application of the method in several previous cases, we faced some challenges for guiding the participants in this case study. These challenges were mainly related to the project management tools that restrained us to reuse all the previously developed data collectors: some project management tools had no data collector deployed in the company and had to be developed beforehand. No other relevant problems were reported.

On the other hand, the neutral score for the self-explanatory aspect of the method was also somewhat expected, mainly because the participants from the company were not requested to read or be formally trained for the method execution. Instead, we participated as experts of the method and guided all the activities and tasks.

Regarding the negative perception about the lack of clarity for choosing the historical data set, we are aware that this is a relevant aspect, as we must clarify the effect of choosing certain timeframes into the resulting model.

All in all, we emphasize the relevance of the received feedback to improve and shape future executions of the method.

Regarding the lessons learnt, we would like to highlight some important aspects from this case study conducted as part of the summative validation of the method:

- The importance of properly communicating the potential benefits of the method to the target company without endangering the message with technical details. We consider that showing the companies the value of the SSI estimation models, their “what-if” analysis capabilities, and the possibility to connect them to the data collectors to provide the SSI monitoring was adequate, as it could engage them to conduct the *Product Readiness* case study. However, we did not provide them with technical details about the BNs and the used tools, with the aim to avoid overwhelming them with excessive details.
- Respecting and adapting the applicability of the method to the company’s time constraints and working rules. For instance, when applying the method in Modeliosoft, we realized that the size and complexity of the specified SSI would have an impact on the time required to build the associated SSI estimation model, as the elicitation of probabilities is a time-consuming step. Therefore, we automated several parts of the process by developing our tools for easing the

elicitation of probabilities. We also had to enhance some parts of the method to tailor it to the types of interactions we had with the participants of the company. In most of the cases, our interaction with Modeliosoft participants was indirect (through emails), so we had to design the mentioned supporting instruments to ease them to provide us the required information at their own pace, without an excessive impact on their daily schedules. As some of the indirect interactions required access to their project's data in order to prepare such supporting assets, we realized that our expectations on the access to the data were too high. Not only in Modeliosoft but also in our previous formative evaluations the data access was difficult, due to confidentiality reasons. Software companies set up secure protocols and mechanisms to provide us access to their data. Due to that reason, we had the extra tasks of adapting our interactions and tools to such data access constraints.

- The need of counting with the right domain experts. The information specified by the experts affects the accuracy of the SSI estimation model. We highlighted this point to the Modeliosoft representative, so he paid special attention to assign suitable personnel. We adapted guidelines and instruments to enable such personnel to fill in the required information considering their specific daily schedules and constraints. Regarding the probability quantification (i.e., filling in the CPTs) of the BN nodes, we found out that ideally, probabilities for each node should be quantified by an expert (ideally a decision maker) on the variable represented by the node, as he/she should be able to quantify the relationships between the involved variables.

Finally, the fact of engaging with industry to get insights of the real application of the method fosters industrial uptake. So far, during the execution of this thesis, we shared some of the preliminary results of this case study with other industrial representatives of the Q-Rapids project and they seemed interested on the applicability of our method with a similar approach as the one followed in Modeliosoft (i.e., in a pilot project under our supervision).

8.8 Threats to validity from the ModelioNG's case study

In this subsection, we present the considered threats to validity from the ModelioNG case study to specify and create an estimation model for the *Product Readiness* SSI. The threats to validity detailed herein from the case study refer to the summative validation of the method, based on our perceptions and the ones communicated by the participants of the case study.

- **Internal validity:** We conducted a participatory case study, where we as researchers played an active role assisting the domain experts from the company in each step of the method and providing detailed explanations or eventually technical help for what information or data collector they had to provide and/or generate. Therefore, this had a great influence on the smooth execution of the steps of the method, as we designed the method and therefore are experts on it. We emphasize that we are aware of that and was part of our strategy for fostering the willingness of the company to participate. Thus, our main insights come from our participatory observations during the case study and the feedback from the company. Other factors that might positively affect the internal validity of the case study, the resulting artifacts, and the perceptions of the participants of the company are: a) domain experts from the company were selected by the Modeliosoft representative without any intervention from us (except for the indications mentioned in Chapter 6), and mainly based on the suitability of the expertise required for each step and task of the method during the case study execution. This is important as their suitability affects not only the execution of the method but also the correctness of the resulting artifacts. We are aware that the selection of the participants could have been affected by the availability of the domain experts in the case study. However, we did not experience any case where the domain experts did not have the required expertise for providing us the required information. Therefore, it seems that all participants were really experts in their corresponding areas.
- **Construct validity:** We followed the SESSI method phases and steps to drive the execution of the case study. In addition, we designed and validated a questionnaire to gather participants' feedback and used our own diaries to register our observations as case study participants. As we mentioned above, an important aspect of this case study was that we adapted as much as possible the

execution of the method to the needs of Modeliosoft. For instance, we prepared additional material and adapted some activities of the method to be performed online instead of face to face with domain experts as this was more convenient for Modeliosoft. Another adaptation was related to the feedback questionnaire, mentioned in section 8.7 (and available in the Appendix 2 (section 2)). It was originally aimed to be answered by each Modeliosoft participant, however, Modeliosoft considered more convenient to fill in a single questionnaire with the agreement from all participants. To deal with this situation, we relied on the provided answers as representative of the general perception of Modeliosoft participants but added a triangulation activity for confirming the results. This triangulation activity was performed by the research team as follows: We set up two main data sources. On the one hand, the first data source resulted from grouping all the relevant observations, insights, and interactions with Modeliosoft collected using our individual diaries, as part of the Data Collection and Data Analysis procedures stated in section 8.3. On the other hand, the second data source consisted in the fulfilled feedback questionnaire received from Modeliosoft. We, as research team, conducted an internal meeting in which, for every part of such fulfilled questionnaire (i.e., closed aspect scores and open questions), we validated that there was not any inconsistency or discrepancy with respect to the first data source. All in all, the design of the case study was quite flexible to deal with contextual situations and we did not experience relevant problems for such adaptations, and we could even reuse some previously developed data collectors.

- **External validity:** The purpose of the conducted case study was not to generalize our observations regarding the method execution but to learn and understand some practical implications of applying our method in a real environment. Therefore, we described the setting and details on the execution of each step of the SESSI method in the case study as much as possible, so our results can be examined. Furthermore, the resulting estimation model and related artifacts should be interpreted with caution, considering that it was built in the specific context of the ModelioNG case study and by the participating members. Therefore, variations on the resulting SESSI method (including its companion tool support) should be expected when adapting the method for other cases (i.e., changing the database storing the historical data, using different data collectors, etc.).

9 SESSI’s Usefulness Insights

This chapter details the last research stage of this thesis, led by **RQ4** “*Is it feasible to use the resulting assets from the SESSI method for enabling advanced decision-making support?”*.

9.1 Research Context and Research Approach

As a result of applying the SESSI method to specify, estimate, and monitor the *Product Readiness* SSI in Modeliosoft (as detailed in Chapter 8), we promoted a further collaboration to explore the feasibility of using the resulting models, data and infrastructure from the SESSI method for enabling advanced decision-making support.

Modeliosoft was interested on advanced decision support based not only on monitoring the actual status of the SSIs but also estimating their future values, i.e., forecasting the values of the SSIs. Forecasting is the process of making predictions of the future, mostly based on past and present data, and trends (Chambers et al., 1971). Monitoring the forecasted values of the SSIs of a software company may provide funded evidence of a potentially high risk or opportunity that could help diverse company roles to anticipate actions accordingly (i.e., to prevent undesired effects and/or promote beneficial states). For instance, forecasting the *Product Readiness* SSI of a software product under development might reveal a likely risk of deadline violations, therefore the allocation of additional resources to avoid such undesired effect can be early promoted by decision makers. Another example could be that forecasting the bug density metric might assist decision makers to prevent releasing software products when the forecasted values of the metric show an uptrend in such bugs.

Hence, based on such interest in forecasting, the goal of the intended study was:

> *"To envisage a suitable and technically feasible forecasting solution based on the resulting assets from the SESSI method. By suitable solution we mean that the SSI forecasting performance should be above an expected threshold stated by Modeliosoft. By technical feasibility we mean that the proposed solution should be successfully operationalized and deployed into the Modeliosoft's testing infrastructure."*

A collaboration team was formed to reach the stated goal. It was composed of three researchers and 2 Modeliosoft's representatives. One of these representatives covered a high strategic role at the company while the other covered a project leader role with high operational knowledge.

We adopted action-research. In general, action research proposes an interactive inquiry process that balances problem-solving, evaluation and learning activities implemented in industry-academia collaborations in order to improve industry practices (Avison et al., 1999; Elden and Chisholm, 1993). The research design plan of this stage was inspired mainly on the action-research cycle of five phases proposed by Susman and Evered (Susman and Evered, 1978), as a backbone: 1) *Diagnosis* (identifying or defining a practical problem). 2) *Action planning* (considering alternative approaches to solve the problem). 3) *Action taking* (setting the planned actions into practice). 4) *Evaluation* (studying the consequences of an action). 5) *Specifying/learning* (identifying findings). Table 15 summarizes the steps of the cycle, their goals, main activities done by the collaboration team and the corresponding main assets produced.

Table 15 Details of the action-research activities, goals and produced assets

Phase	**Goal**	**Main Activities**	**Main Produced Assets**
Diagnosis	To elicit Modeliosoft's forecasting needs.	-Discussion sessions -Semi-structured interviews	**Set of forecasting requirements**

Action Planning	To design a blueprint of the forecasting solution for Modeliosoft using the assets from the SESSI method	-Brainstorming sessions -Hands-on sessions to develop software supporting tools	**Blueprint** of a forecasting solution based on the SESSI method resulting assets and infrastructure
Action Taking	To apply the forecasting solution in a pilot project in Modeliosoft.	-Design and execution of a case study to apply the forecasting solution in a Modeliosoft's pilot project	**Results** about: **-Suitability** **-Technical feasibility of the forecasting solution**
Evaluation	To evaluate the forecasting solution	-Feedback evaluation	**Feedback evaluation**
Specify Learning	To identify relevant observations and lessons learnt.	-Discussion sessions based on the observations from the case study	**Lessons learned**

9.2 Diagnosing: Modeliosoft Forecasting Requirements

Apart from reusing the resulting assets from the SESSI method, Modeliosoft's forecasting needs were elicited from discussions with Modeliosoft's representatives. They together provided a comprehensive strategic and operational view of the expected forecasting requirements. In addition, to gather the requirements from other key roles of the company that might use the forecasting capabilities, we analysed semi-structured interviews reported at (Q-Rapids, 2018a). The set of main high-level forecasting requirements are summarized in Table 16.

Table 16 Modeliosoft's requirements for the forecasting solution

Requirement	**Description**
R1	To have an automated solution for helping several roles of the company (e.g., CEO, project leaders) to better inform their decisions based on the forecasting values of SSIs.
R2	To minimize the specialized knowledge required for putting forward a forecasting solution. It is because most employees do not have extensive data mining knowledge. The idea is that the solution is adapted as much as possible to the technical background of project leaders of the company (one of the representatives

	covered such role), so that they are able to put forward the solution in their corresponding projects with minimal support.
R3	To define an expected performance threshold according to the specific needs of the project and SSI. Modeliosoft's representatives stated that in general, Modeliosoft could assume 70% as the forecasting performance threshold for those projects that are not in critical situation but for those in critical situation a threshold above of 80% would be expected.

After discussing several alternatives in informal brainstorming sessions, the collaboration team chose one that offered a straightforward way of automatically forecasting the values of SSIs and maximized the reuse of existing assets from the SESSI method as well as the coverage of Modeliosoft's forecasting requirements.

It consisted on using the SSI Estimation Models produced by the SESSI method as the backbone for building SSI Forecasting Models for forecasting the values of the SSIs. This was considered the most convenient alternative because:

- There was in-house know how about building estimation models for any relevant SSI required by Modeliosoft.
- The existence and previous use of the SSI Estimation Models ensured the availability of historical data about estimations of the SSI as well as the availability of data collectors. Therefore, this maximizes the reuse of Modeliosoft's infrastructure.
- The current Bayesian Network (BN)-based structure of the SSI Estimation Models used in Modeliosoft already deals with potential missing values and explainability aspects. This was crucial to minimize the complexity of reaching a suitable forecasting solution because incomplete data and explainability issues are two of the most critical problems of forecasting in real world (Schelter et al., 2018).
- Constructing the SSI Forecasting Model based on a similar BN-based structure than the SSI Estimation Models ease the reuse and integration of new components into Modeliosoft's dashboard in order to visualize the forecasting results.

Having all these ideas in mind, the collaboration team produced a blueprint of the forecasting solution.

9.3 Action Planning: Blueprint of a Forecasting Solution

To realize how to put forward the selected solution after the brainstorming sessions, the researchers did hands-on sessions aimed to technically try out the construction of SSI Forecasting Models. We were supported by Modeliosoft's representatives, specially one of them (the one covering the role of project leader) as he has technical knowledge and helped us to properly shape the resulting processes and tools.

The resulting blueprint was applied to a pilot project tackled as a case study presented below. The subsections below provide details of the two prescriptive phases of the blueprint of the forecasting solution.

9.3.1 Phase 1: Building and Evaluating an SSI Forecasting Model

The main idea behind the construction of the SSI Forecasting Model is to use the existing SSI Estimation Model as a backbone. Based on the BN nature of the SSI Estimation Model, its corresponding metric nodes are fed with the forecasted values of each metric. The aim is to propagate such values up to the BN and get the forecasted states and probabilities for all nodes up to the SSI.

To forecast the values of the metrics, additional models/databases should be built using the historical data available.

Figure 26 provides a summary of inputs and outputs of Phase 1. The required steps to build and evaluate the SSI Forecasting Model are detailed below.

Step 1 and 2 require information that should be defined by a role that understands why and for what the intended SSI Forecasting Models is needed.

Step 1: *To define the forecasting horizon and the corresponding training and validation sets*. The forecasting horizon defines the length of time into the future for which forecasts are to be computed. Once the forecasting horizon has been defined, the very first action is to split the available SSI estimation historical data into training and validation sets. The selection of suitable training/validation splits is quite important because as more random variation in the historical data, more data will be needed for obtaining forecasting models that suitably capture such variations (Hyndman and Kostenko, 2007). In other words, the more training data is available, the higher the potential performance of the forecasting (Armstrong, 2001). So, suitable splits depend on the amount of available SSI estimation historical data and the desired forecasting horizon. Commonly used splits are 70%-30% or 80%-20% for training and validation sets, respectively (Raschka, 2018). Hyndman and Athanasopoulos (Hyndman and Athanasopoulos, 2018) also recommend that the validation set should be at least as large as the maximum forecasting horizon.

Step 2: *To define the expected performance threshold*. It refers to the extent to which the intended SSI Forecasting Model predicts well-founded values. For quantifying performance, there are several potential metrics that could be used according to the context and objective of the decision makers. A set of typical performance metrics such as Accuracy, Precision, Recall and F1 score metric (Galdi and Tagliaferri, 2018) can be applied. Other metrics such as Jensen-Shannon distance (Endres and Schindelin, 2003) and metrics' contribution error were also included to provide additional information for supporting the decisions related to the need of improving the quality of the SSI Forecasting Model.

Accuracy is a metric that generally describes how the model performs across all the states. It is defined as the ratio between the number of correct predictions with respect to the total number of predictions. It is easily interpretable and useful in balanced data. Precision is defined as the proportion of true positives on the total number of predicted positive instances (Galdi and Tagliaferri, 2018). It is useful when it is important to reduce the number of false positives. Recall, which is defined as the proportion of true positives on the total number of actual positive instances (Galdi and Tagliaferri, 2018). It is useful in applications where it is important to reduce the number of false negatives. F1 score (also known as F-measure) combines the recall and precision metrics as their harmonic mean (Galdi and Tagliaferri, 2018). It is useful when both precision and recall are important and a balance between both is preferred.

On the other hand, the Jensen-Shannon distance metric (Endres and Schindelin, 2003) measures the distance between the probability distributions of the forecasted states inferred by the SSI Forecasting Model and those from the validation set. A value close to zero means that the probability distributions are quite similar thus confirming the suitability of the SSI Forecasting Model. The metrics' contribution error metric aims to quantify the relative importance of the metric nodes (only those that have a forecasting model) together with their potential prediction error. It is calculated by averaging the difference across the validation set between the Jensen-Shannon distance metric value of the SSI node versus the corresponding Jensen-Shannon distance metric when feeding only individual forecasted metric’s values (i.e., using the prior probabilities of the model for the other metric nodes).

While the Jensen-Shannon distance metric helps to confirm the confidence on the accuracy of the forecasted values of the nodes, the metrics’ contribution error provides insights on the relative contribution of the node to the potential forecasting errors of the SSI.

We developed a software tool called *Accuracy Computation* and *Distance Computation* (see Table 17) for supporting the calculation of some performance metrics.

In line with **R3**, we can consider that the minimum expected performance for critical projects is 80% and for non-critical ones is 70%. However, each project could define its own expected performance above such minimal values.

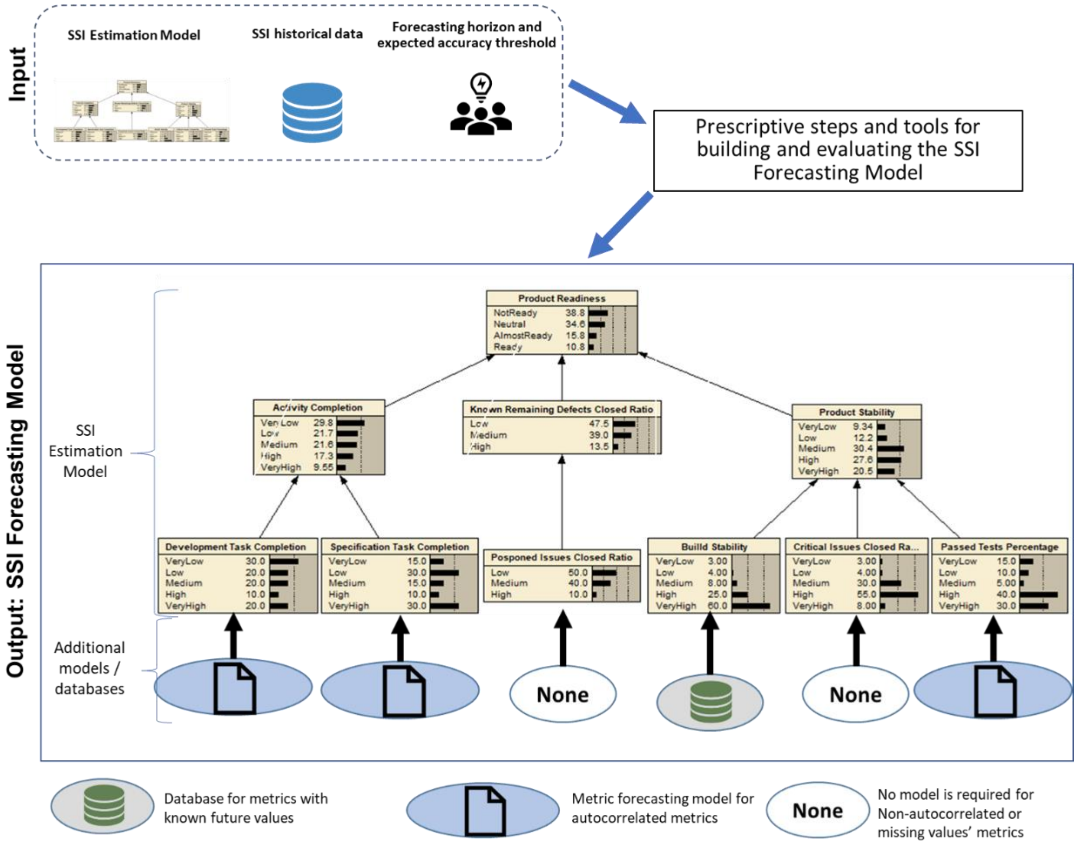


Figure 26 Summary of Inputs and Outputs of Phase 1

Step 3: *To forecast the metrics’ values corresponding to the SSI Estimation Model*. To determine how to forecast the metric values, the training set is used. Three different types of metrics were devised. For each type, adequate mechanisms to forecast them for the given forecasting horizon were defined. The classification is as follows:

a) **Metrics with Known Future Values:** It refers to metrics for which their future values for the forecasting horizon are known or can be estimated by expert knowledge. For instance, thresholds or values related to the allocation of resources. For this type of metrics, the known or estimated values for the forecasting horizon are stored (for example, in a database) and used as forecasts.

b) **Non-Autocorrelated or Missing Metrics Values:** It refers to metrics that do not show any autocorrelation in the training data (i.e., no past values can predict their future behaviour) and/or do not have historical data available for computing their forecasts. To support the identification of non-autocorrelated metrics, we developed the *Autocorrelation Test* software tool (see Table 17) that allows the graphical visualization and/or numerical analysis of a time series dataset to assess its autocorrelation. The best way to get the forecasted values of non-autocorrelated and missing metrics is to rely on domain's expert judgments. However, if it is not possible, such values can rely on the prior probabilities of the corresponding metric nodes from the SSI Estimation Model. Note that these prior states ultimately come from domain experts that built the SSI Estimation Model, as explained in Chapter 6.

c) **Metrics with Autocorrelated Values**: It refers to metrics that show autocorrelation in the training data (i.e., the past values of the time series can be used as predictors). Forecasting metrics with autocorrelated values based on the training data led to a time-series forecasting problem. A software companion tool that generates time-series models was developed (see *Time Series Gathering* tool in Table 17). We aimed to find a suitable forecasting model for each metric of this type. The process for finding a forecasting model for each autocorrelated metric is described in Step 4.

Table 17 Tools for supporting and automating processes of Phase 1

Step	Software tool description	Input	Output
3	**Time Series Gathering Tool** (Manzano, 2021a):	-Database parameters -Dataset -Frequency	-Dataset modelled as a time series

	Generates a time series model in R (Development Core Team, 2008) from a dataset extracted from a database[35].	-Training period	
3	**Autocorrelation Test Tool** (Manzano, 2021b): Assess autocorrelation graphically and numerically based on (Hyndman and Khandakar, 2008) and (Hyndman and Athanasopoulos, 2018).	-Time series dataset	-Graphical/ numerical confirmation of autocorrelation.
4	**Hold-Out Approach Kit:** It wraps and uses Model Fitting, Forecasting Execution, and Model Comparison tools for automatically applying the hold-out approach.		
	Model Fitting Tool (available for the forecasting techniques shown in Table 18) (Manzano, 2021c): Fits a forecasting technique with a time series training dataset to obtain a fitted model with the selected technique.	-Training dataset -Forecasting technique	-Fitted model
	Forecasting Execution Tool (Manzano, 2021d): Executes a forecasting model with a given forecasting horizon to obtain the corresponding forecasted values.	-A forecasting model -Forecasting horizon	-Forecasted values
	Model Comparison Tool (Manzano, 2022c): Compares forecasting models and selects the best one based on accuracy and Root Mean Squared Error metrics.	-Validation set -Forecasting models	-Most accurate forecasting model
5	**Accuracy Computation Tool** (Manzano, 2022d): Computes the accuracy of the SSI Forecasting Model.	-Validation set -SSI Forecasting Model	-Accuracy of the SSI Forecasting Model
5	**Distance Computation Tool** (Manzano, 2021e): Computes the Jensen-Shannon distance (Endres and Schindelin, 2003; Lin, 1991) between two probability distributions datasets.	-Validation set -Forecasted states of the SSI	-Average distance between the forecasts and the validation set

[35] It uses an Elasticsearch database but can be easily changed to other technology

5	**Forecasting Report Tool** (Manzano, 2022e): Generates a CSV report from the results obtained from the tools above.	-Results from the tools used in the previous steps	-Comprehensive report

Step 4: *To train and select the best forecasting model for each autocorrelated metric.* To promote the use of the intended forecasting solution in diverse projects in Modeliosoft, we decided to automate the use of a heterogeneous set of forecasting techniques. Table 18 summarizes the set of forecasting techniques considered for forecasting the values of autocorrelated metrics. This set of forecasting techniques aims to cover the 3 main families of time series forecasting techniques with automated R implementations: 1) *statistical forecasting techniques* such as ARIMA, exponential smoothing and decomposition models, 2) *advanced forecasting techniques* such as Hybrid forecast models (Shaub and Ellis, 2020) and Bagged ETS (Bergmeir et al., 2016) and 3) *machine learning-based forecasting techniques* such as neural networks. The importance of including at least one technique from each family was based on the fact that there is not a single forecasting technique providing the best results for every situation. So, covering the main families of forecasting techniques helps to maximize the chances of finding an adequate forecasting model for any metric from Modeliosoft's projects.

Table 18 Forecasting techniques used to devise metrics' forecasting models

Type	**Acronym**	**Full Name/Reference**
Statistical	ARIMA	Autoregressive Integrated Moving Average (Newbold, 1983)
	ARIMA FS (forcing seasonal models)	Autoregressive Integrated Moving Average forcing seasonal models (Newbold, 1983)
	THETA	Theta model: a decomposition approach to forecasting (Assimakopoulos and Nikolopoulos, 2000)
	ETS	Exponential Smoothing State Space Model (Hyndman et al., 2008)
	ETS DM (forcing damped models)	Exponential Smoothing State Space Model forcing damped models (Hyndman et al., 2008)
	STL	Seasonal Decomposition of Time Series by Loess (Cleveland et al., 1990)

	TBATS	TBATS model (Exponential smoothing state space model with Box-Cox transformation, ARMA errors, Trend and Seasonal components) (De Livera et al., 2011)
Advanced	BAGGED ETS	Bagged ETS (Bergmeir et al., 2016)
	HYBRID	Hybrid Forecast Models (Shaub and Ellis, 2020)
	PROPHET	Prophet (Taylor and Letham, 2018)
Machine Learning	NN	Feed Forward Neural Networks (Hyndman and Khandakar, 2008; Tadeusiewicz, 1995)

To automate the selection of the best forecasting model for each metric, we implemented the *Hold-out Approach* software kit (see Table 17) that operationalizes the hold-out approach suggested by Cerqueira et al. (Cerqueira et al., 2020). Figure 27 provides an overview of the implemented hold-out approach. This approach consists of two main processes:

1) Fitting a candidate forecasting model from each forecasting technique presented in Table 18 using the training set.
2) To select the best forecasting model. To do so, the set of candidate forecasting models are used to forecast the metric values for the same period as the validation set. These forecasted values are then compared with the metric values from the validation set.

A composite criteria combining two metrics was used to select the best forecasting model: a) accuracy i.e., the percentage of matches between the forecasted metrics and the metric states from the validation set; and b) the Root Mean Squared Error (RMSE) metric, as it is suggested as a suitable metric to compare results from time series forecasting models that are in the same numerical scale (Hyndman and Athanasopoulos, 2018; Hyndman and Koehler, 2006). The optimal RMSE metric is zero (meaning that there is no forecasting error). Using this composite criterion, the best forecasting model will be the one yielding the highest accuracy and the lowest RMSE value (prioritizing the accuracy).

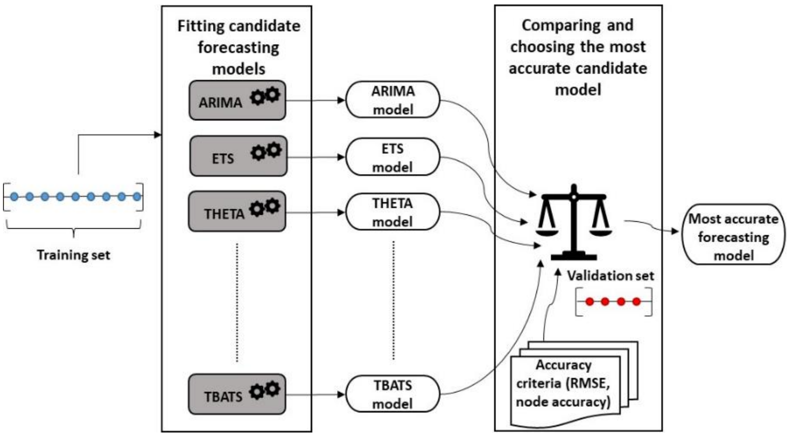

Fitting candidate forecasting models
ARIMA
ETS
THETA
TBATS
ARIMA model
ETS model
THETA model
TBATS model
Training set
Comparing and choosing the most accurate candidate model
Validation set
Accuracy criteria (RMSE, node accuracy)
Most accurate forecasting model

Figure 27 Overview of the hold-out approach implemented in this work

It is important to remark that the set of 11 forecasting techniques and the evaluation metrics included so far in the *Hold-Out Approach* can be extended in the future to deal with the forecasting needs of Modeliosoft's projects.

Step 5: *To articulate the SSI Forecasting Model and its performance evaluation*. This step aims to articulate the SSI Forecasting Model and confirm that its performance fulfils the threshold defined by the company in Step 2.

Once the corresponding models/databases to be used to forecast each metric have been built, they should be articulated to compose the SSI Forecasting Model. Thus, the SSI Forecasting Model is composed by the SSI Estimation Model and the additional models/databases for each corresponding metric, as stated in Figure 26. The SSI Forecasting Model provides the SSI forecasting data (i.e., the forecasting of the SSI, that includes all the states of their composing nodes and their probabilities, as well as the forecasted metric values).

The performance evaluation of the resulting SSI Forecasting Model is based on the performance metrics defined in Step 2. It is expected that the computed performance is equal or above the expected performance threshold defined by the company.

In case the expected performance is not achieved, several potential actions should be discussed in order to decide whether:

- To relax the stated performance threshold to proceed with the following phase.
- To promote the repetition of the previous phases with the aim of improving it. For instance, increasing the set of historical data, to adjust the forecasting horizon so that the amount of historical data considered for training the models

increases (Hyndman and Athanasopoulos, 2018), or to add new forecasting techniques to the *hold-out approach kit*.

The metrics' contribution error could also provide some insights about which metrics are relatively contributing to the errors of the SSI Forecasting Model.

To support the evaluation of the SSI Forecasting Model, we developed a *Forecasting Report* software tool (see Table 17) that generates a comprehensive report of meaningful details of the tool-supported performance metrics.

9.3.2 Phase 2: Deploying the resulting SSI Forecasting Model into Modeliosoft's Dashboard

To deploy the SSI Forecasting Model and enable its integration into the Modeliosoft's dashboard, we provide the description of a high-level architecture, based on the reuse of Modeliosoft's dashboard current architecture (Figure 24).

We suggest the implementation of the following components as services:

- *SSI Forecasting* component. It receives as input a forecasting period and returns the SSI forecasting data for the given forecasting period. It encapsulates other components:
    - *SSI Forecasting Wrapper* component. It receives a forecasting period and sends a request to the *SSI Forecasting Computation* component for calculating the SSI forecasting data for the given period.
    - *SSI Forecasting Computation* component. It is in charge of calculating the SSI forecasting data for the given period. To do so, it queries the corresponding *Additional Databases* that contain data from Metrics with Known Future Values and/or Metrics with Missing values provided by domain experts. The *RServe API* is in charge of obtaining the forecasted values of autocorrelated metrics from the *Files of Metrics' Forecasting Models*. Once the forecasted metric values have been obtained and discretized to metric states, they are sent to the *SSI Estimation Computation* component to feed the corresponding SSI Estimation Model and return the SSI forecasting data.

To integrate the *SSI Forecasting* service into the existing infrastructure, we suggest to extend the existing *Dashboard Backend* component with a *Forecasting Functionality* component in charge of orchestrating the functionality related to forecasting. Figure 28 shows a high-level sketch of a potential integration of the *SSI Forecasting* service into the current Modeliosoft's dashboard.

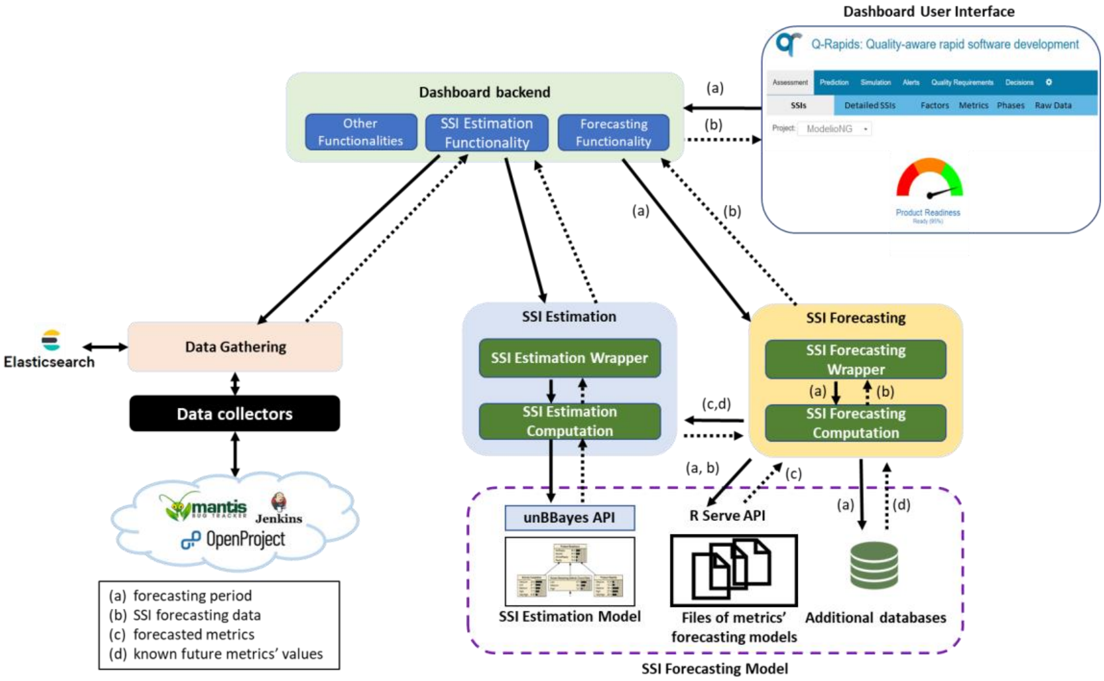


Figure 28 High-level sketch of a potential integration of the proposed forecasting solution into the Modeliosoft's dashboard

There is evidence that this high-level architecture worked reasonably well for enabling SSI forecasting in the context of the pilot project presented in section 9.5. Therefore, it is considered as a reference architecture for Modeliosoft's SSI's forecasting projects. We remark that each SSI's forecasting project has its own needs and characteristics (e.g., expected forecasting demand, amount of data involved, etc.). These needs and characteristics widely affect the architectural and deployment related decisions. These decisions are not further discussed here as they are context dependent. For instance, in cases where the forecasting demand and the data involved are quite high, it can be considered that the *Dashboard Backend* component manages the store/materialization of the computed metrics' forecasted values into a database in order to improve the performance of subsequent requests.

9.4 Action Taking

To evaluate the suitability and technical feasibility of the forecasting solution, it was tried it out in a pilot project. We tackled the pilot project as a case study and followed the guidelines suggested by Runeson et al. (Runeson and Höst, 2009).

9.4.1 Case Study Design

The key research question leading the case study was:

> *"How is the application of the forecasting solution in the studied project?"*

We were especially interested on gathering evidence on its suitability (i.e., the performance of the resulting SSI Forecasting Model is above the expected threshold stated by Modeliosoft) and technical feasibility (i.e., the proposed forecasting solution should be successfully operationalized and deployed into the Modeliosoft's testing infrastructure).

The main drivers for the case study design were the phases and steps suggested as the forecasting solution, as well as the specific characteristics and needs of Modeliosoft.

The case study design was flexible to deal with potential unexpected issues. Details of issues and/or decisions taken during the case study are detailed in the execution section.

Modeliosoft's representatives selected Modelio Wyrm, as a suitable pilot project to apply the forecasting solution. The selection of the project was mainly opportunistic and based on the following factors:

1) Modeliosoft had an interest on forecasting the *Product Readiness* SSI for this project and were open to share some of their SSI estimation historical data to run a pilot project.
2) The project had available historical data about the *Product Readiness* SSI (i.e., the estimations of the SSI, that includes all the states of their composing nodes and their probabilities, as well as the metrics' values).
3) The project was being monitored using the existing *Product Readiness* SSI Estimation Model.

Points 2 and 3 are actually a requirement to apply the forecasting solution as they promote the reuse of assets from the SESSI method.

Three researchers and one Modeliosoft's representative who are also members of the collaboration team participated directly in the case study. Such Modeliosoft's representative was also the project leader of the Modelio Wyrm project and he acted as our contact point for the execution of this case study. In the context of this case study he will be referred as the Modelio Wyrm project leader. He provided us with all required information about Modelio Wyrm and helped us to shaped all software tools (when needed) to fit them to the pilot project needs. Other Modeliosoft's employees related to Modelio Wyrm also participated in the application of the forecasting solution to Modelio Wyrm but had an indirect contact with the researchers (mainly because of covid-related restrictions).

All issues during the execution of the pilot project were recorded and discussed among all participants and subsequently assessed and reflected in the corresponding Specifying Learning activity of the industry-academia collaboration.

9.4.2 Modelio Wyrm Project

Modelio Wyrm is the codename of the version 4.0 of Modelio software. Its development process is analogous to the ModelioNG's presented in Chapter 8. The development team used the same management tools: *OpenProject*[36] for project management (backlog management, issues, and specification tracking), *Mantis*[37] for bug tracking and *Jenkins*[38] for building and testing.

Modelio Wyrm is a strategic project for the company and its *Product Readiness* SSI was being monitored and visualized through the corporate dashboard. It implies that there exists a *Product Readiness* SSI Estimation Model, the corresponding data collectors and historical data available.

9.4.3 Data collection and Data Analysis

For security and confidentiality reasons, we did not have direct access to the company repositories, but the Modelio Wyrm project leader provided us with a snapshot of 47 days of historical data about the *Product Readiness* SSI, corresponding to a specific

[36] https://www.openproject.org

[37] https://www.mantisbt.org

[38] https://jenkins.io

release under development of Modelio Wyrm. Such data was obtained automatically from data collectors. These data collectors extracted and stored information from Modeliosoft’s corporate repositories (mainly development process/product-related repositories) on a daily basis. The data was stored in an Elasticsearch[39] node.

Data analysis was performed according to the phases and steps of the forecasting solution. We also used individual diaries to record notes on any type of issues and aspects that we considered relevant (e.g., problems, schedule, effort, decisions, attitudes). Improvements and adaptations done to the software tools together with their rationale were also recorded. We provide as much detail as possible given non-disclosure agreements with Modeliosoft.

In addition, we designed a survey based on a questionnaire as a data collection instrument for gathering feedback from Modeliosoft’s employees that participated in the pilot project about the technical feasibility of the forecasting solution (see details in section 9.5).

9.4.4 Case Study Execution

The project leader of Modelio Wyrm involved 5 people from his Modelio Wyrm’s team in the execution of the pilot project. He wanted them to have first-hand contact with the forecasting solution in order to know their impressions about it after the execution of the pilot project. A general explanation of the phases and steps of the proposed forecasting solution was given to them before the execution of the case study. They were also informed about goals of the pilot project and some important aspects regarding the forecasting solution. The provided explanation was general and tried to avoid technical details.

Phase 1: Building and Evaluating the *Product Readiness* Forecasting Model

Step 1: *To define the forecasting horizon and the corresponding training and validation sets*. Based on the available historical data and the forecasting needs of the project, the forecasting horizon was set to 14 days. The Modelio Wyrm project leader stated that this forecasting horizon was meaningful to provide enough room to prevent risk

[39] https://www.elastic.co

situations for the specific release under development of Modelio Wyrm. The SSI estimation historical data was split into 33 days for the training set (70% of the total) and the remaining 14 days (corresponding to the forecast horizon) were used for the validation set (30% of the total).

Step 2: *To define the expected performance threshold*. It was set to 70%. Modelio Wyrm's project leader confirmed that the project was not in a critical situation at such moment, so he decided to assume 70% as the forecasting performance threshold.

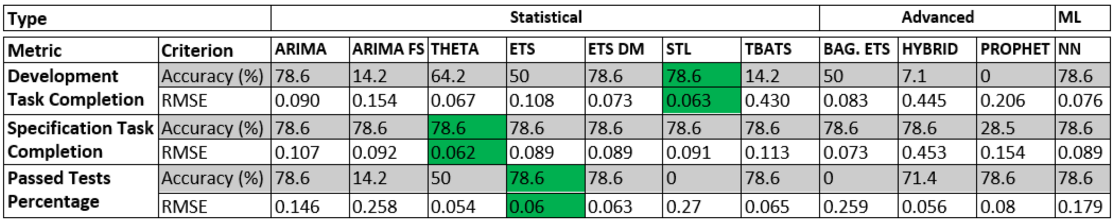
Step 3: *To support the classification of metrics of the Product Readiness Estimation Model.* The metrics' values of the training set were modelled as standard time series in R, using the software tool *Time Series Gathering*. The *Autocorrelation Test* tool helped to confirm if it was autocorrelation or not. Table 19 summarizes the classification of each metric of the *Product Readiness* Estimation Model and its rationale.

Table 19 Classification of Metrics, Rationale and Storage of Forecasted Values

Categorization	**Metric**	**Rationale**	**Obtention and/or Storage of Forecasted Values**
Metric with Autocorrelated Values	-Development Task Completion -Specification Task Completion -Passed Tests Percentage	Their autocorrelation was confirmed by the information provided by the *Time Series Gathering* and the *Autocorrelation Test* tools.	The forecasted values are obtained through their corresponding forecasting model (to be built in step 4 below).
Missing Metric's Values	-Postponed Issues (Closed) Ratio -Critical Issues (Closed) Ratio	In both cases there was no data for the period stated in the training data. They were not collected during the training period for an unexpected problem in the corresponding data collectors.	Modelio Wyrm project leader decided to get their forecasted values from the prior *Product Readiness* SSI Estimation Model's probabilities, therefore no values were stored.
Metric with Known Future Values	-Build Stability	Modelio Wyrm project leader confirmed (after discussing with the development team) that the expectation for this metric would be to keep its constant value for the forecasting horizon (i.e., any change was expected in the percentage of successful builds with respect to the total of builds triggered in each seven-day period).	The constant value for the forecasting horizon was stored in a physical storage (e.g., databases, CSV files).

Step 4: *To train and select the best forecasting model for each autocorrelated metric*. For each autocorrelated metric, a set of forecasting models (corresponding to the 11 techniques currently implemented in the *Hold-Out Approach kit*) were trained and evaluated. Table 20 summarizes the resulting values of the accuracy and RMSE metrics for each one of the models built. In green it is shown the selected model for each metric (i.e., the one that had the best results from accuracy and RMSE).

Table 20 Results of the hold-out approach for each autocorrelated metric



Type		Statistical							Advanced				ML
Metric	Criterion	ARIMA	ARIMA FS	THETA	ETS	ETS DM	STL	TBATS	BAG. ETS	HYBRID	PROPHET	NN
Development Task Completion	Accuracy (%)	78.6	14.2	64.2	50	78.6	78.6	14.2	50	7.1	0	78.6
	RMSE	0.090	0.154	0.067	0.108	0.073	0.063	0.430	0.083	0.445	0.206	0.076
Specification Task Completion	Accuracy (%)	78.6	78.6	78.6	78.6	78.6	78.6	78.6	78.6	78.6	28.5	78.6
	RMSE	0.107	0.092	0.062	0.089	0.089	0.091	0.113	0.073	0.453	0.154	0.089
Passed Tests Percentage	Accuracy (%)	78.6	14.2	50	78.6	78.6	0	78.6	0	71.4	78.6	78.6
	RMSE	0.146	0.258	0.054	0.06	0.063	0.27	0.065	0.259	0.056	0.08	0.179

The obtained results were quite positive. We did not have any problem related to finding suitable forecasting models. In all cases, the obtained accuracy of the selected models was higher than the expected performance threshold (i.e., 70%). In addition, the RMSE values for the chosen forecasting models also provided a good fit (accurate forecasts yield RMSE values tending to 0). This implies that the forecasting techniques currently included in the implementation of the *Hold-Out Approach kit*, reasonably covered the metrics of the SSI studied in Modelio Wyrm. In addition, we observed that *statistical forecasting techniques* such as decomposition models (i.e., STL for the *Development Task Completion*, THETA for the *Specification Task Completion* and ETS for the *Passed Tests Percentage* metrics) yielded better results than the *machine learning-based forecasting techniques* and *advanced forecasting techniques*. These results are in line with previous literature findings stating that: 1) *statistical forecasting techniques* are able to provide better forecasts for short time series (as it is the case study of Modelio Wyrm) (Makridakis et al., 2018; Makridakis and Hibon, 2000) and 2) complex forecasting techniques (i.e., *machine learning-based* and *advanced forecasting techniques*) do not necessarily provide the most accurate forecasts for every forecasting problem (Green and Armstrong, 2015; Makridakis and Hibon, 2000).

Step 5: *To articulate the SSI Forecasting Model and its performance evaluation.* The forecasting models/databases for each corresponding metric were integrated with the *Product Readiness* Estimation Model to obtain the resulting *Product Readiness*

Forecasting Model (details of its implementation as a software service are provided in Phase 2).

To support the assessment of the resulting performance, we provided the Modelio Wyrm project leader with a comprehensive report generated by the *Forecasting Report* tool. Table 21 summarizes the results obtained by the software tools. The accuracy of each node is obtained by calculating the percentage of matches between the forecasted states of the node vs. the estimated states of the node for the validation period (14 days). For instance, the *Development Task Completion* node shows a match on 11 out of 14 days. All individual nodes fulfil the expected accuracy threshold stated by Modeliosoft (70%) and the whole accuracy of the SSI Forecasting Model (i.e., the accuracy of the *Product Readiness* SSI node) was 71.43%.

Due to the short length of the validation set, not all the states of the nodes appeared in the dataset. This compromised the use of Precision and Recall (denominators were equal to zero). So, we calculated only F1 score, omitting the missing data. The results were also above the performance threshold for all the nodes.

Furthermore, the values of the Jensen-Shannon distance metrics of all nodes are quite low. This confirms that the distances between the forecasted states inferred by the SSI Forecasting Model and those from the validation set do not compromise the results of the performance metrics.

The results of the computation of the metrics' contribution error for each metric node is also shown in Table 21. It can be observed that the metric node *Development Task Completion* got the highest value for this metric (0.35). It means that this is the metric node with highest relative importance in the whole model or it is the most prone to error. However, no additional actions were required because the accuracy and F1 score of the individual nodes and the *Product Readiness* SSI node were above the expected performance. However, in cases where the performance metrics of the individual nodes and/or those of the SSI are compromised, the value of the metrics' contribution error metric can help to support the decision about which metric's forecasting model should be revised and refit for improving the performance of the SSI Forecasting Model.

All in all, the resulting SSI Forecasting Model for the *Product Readiness* SSI got promising results regarding its suitability. The operationalization of phase 1 took about 6 hours.

Table 21 Results of the evaluation of the SSI Forecasting Model and its nodes

Result/ Node Name	**Dev. Task Compl.**	**Spec. Task Compl.**	**Passed Tests Percent.**	**Activity Compl.**	**Known Rem. Defects (Closed) Ratio**	**Product Stability**	**Product Readiness**
Node Type	Metric			Factor			SSI
Matches (forecasted states vs validation set)	11/14	11/14	11/14	11/14	10/14	12/14	10/14
Node accuracy (%)	78.57	78.57	78.57	78.57	71.43	85.71	71.43
F1 score (%)	88	88	88	88	83	100	83
Jensen-Shannon distance	0.18	0.18	0.18	0.07	0.11	0.04	0.05
Metric Contribution Error	0.35	0.17	0.09	-	-	-	-

Phase 2: Deploying the *Product Readiness* Forecasting Model and Integration with the Corporate Dashboard

After the execution of the previous phase and the successful results got with respect to the suitability of the *Product Readiness* Forecasting Model, the *Product Readiness* Forecasting Model was ready to be deployed. Following the blueprint of the architecture illustrated in Figure 28, most of the core architectural elements already existed in the company, with the only exception of the *SSI Forecasting* service that is in charge of providing forecasting capabilities to the Modeliosoft’s dashboard.

As a part of this pilot project, we developed the *SSI Forecasting Computation* component of the *SSI Forecasting* service. It was built in Java and relies on R scripts (through an RServe connection) for obtaining the forecasted values of autocorrelated metrics, and on the use of CSV files to get stored metrics’ forecasted values (i.e., metrics with known future values and/or metrics with missing values that are estimated by domain experts). All metrics’ forecasted values are then used as input for the *SSI Estimation* Component, which uses the *UnBBayes* API to feed the corresponding SSI Estimation Model and returns the SSI forecasting data. The *SSI Forecasting Computation* component was developed as open source (Manzano, 2022f) and is freely available.

As this was a pilot project, we did not have access to integrate all the components in the production environment of Modeliosoft, instead this forecasting solution was executed and tested in the testing environment of the company. This environment offers a reliable setting similar to the production environment, so we count with evidence of the successful deployment and integration of the forecasting solution in Modeliosoft's current architecture. There was not any relevant integration problem and the Modelio Wyrm team was able to visualize the forecasting functionalities through the Modeliosoft' dashboard.

9.5 Evaluation

During the execution of the pilot project, we took notes of any type of issues and aspects that we considered relevant (e.g., problems, schedule, effort, decisions, attitudes) to improve the proposed forecasting solution. Improvements and adaptations done to the supporting software tools together with their rationale were also recorded.

In addition, we designed an interview-guide instrument aimed to gather the perceptions of Modeliosoft's employees that participated in the pilot project. We used the guidelines stated by Oates (Oates, 2006). Special attention was paid to inquiry on the positive and negative perceptions about the forecasting solution in order to improve it for subsequent action research cycles. The guide included mainly 3 open questions related to a) positive/negative perceptions about the forecasting solution and its execution, b) suggestions and improvements and c) additional comments. We planned to perform semi-structured interviews with 5 Modeliosoft's employees that participated directly on the activities related to the pilot project, but given some organizational restrictions due to Covid-19, we replaced our interview plan. We finally adapted the interview-guide instrument to be used as a guide for a focus group session led by the Modelio Wyrm project leader. He has previous experience on this type of sessions, so he did not require additional training in advance, only reminders and recommendations. We recommended some actions to mitigate the participants' evaluation apprehension. For instance, remarking at the beginning of the session, that the goal of the case study and the focus group was not to evaluate the employees but to assess the feasibility of the proposed forecasting solution and to improve it. Therefore, their honest opinions (without enhancing/hiding issues) were expected.

The five employees that participated closely on the activities of the pilot project attended to the focus group. The Modelio Wyrm project leader provided us with his notes from the focus group. We are aware of the threats associated to the fact that we did not participate directly on the focus group, as it increases the threats of using subjective and not contextualized opinions. However, the feedback received seems quite honest implying positive aspects and issues to be improved that also confirm several of our own notes taken during the execution of the case study.

Subsequently, all collected information was organized and a meeting with the Modelio Wyrm project leader (who is also one of the Modeliosoft's representatives) was planned. We aimed to assess, discuss and consolidate the results. As a consequence of such meeting, we agreed and grouped our main conclusions from the case study as follows:

- *Usefulness of the existing infrastructure and assets from the SESSI method is considered as the most positive aspect of the solution*. The use of the existing Modeliosoft's infrastructure was mentioned as a very positive aspect as they were already familiar with it. On the one hand, the fact that the integration of the resulting models into their existing infrastructure was already studied and the components were properly encapsulated to ease such integration was positively valued by the employees in charge of infrastructures. On the other hand, the fact that the Modeliosoft's dashboard could be also used to consult the forecasting of SSIs was considered a factor that would promote the rapid adoption of the forecasting functionalities. This is because it does not require additional training (as they are already familiar with the Modeliosoft's dashboard usage).
- *Straightforward forecasting solution.* The steps described to put forward the forecasting proposal were mainly perceived as straightforward and feasible by Modeliosoft's employees that participated closely on the activities of the pilot project. One of them commented that the steps described in the solution could be repeatable in other projects as long as they have all the supporting software tools and the potential availability of an expert to guide/support some of their decisions in case they need it. From our notes during the execution of the pilot project, we experienced several questions and doubts mainly related to the meaning of the SSI Forecasting Model (Phase 1) as most of employees did not have previous knowledge

about forecasting methods. It uncovered the need of promoting a basic training program in the company, so the corresponding employees are aware of basic concepts (e.g., how to apply and interpret the performance metrics) that help them to better understand and apply the proposed solution.

- *Availability of software tools supporting the steps of the forecasting solution was the most valued aspect.* Employees agreed that without the supporting tools, they would not be able to forecast SSIs. Employees mentioned that the availability of the software tools saves extensive manual effort and significantly hides the complexity of building a forecasting model. Our notes taken during the execution of the pilot project also support this perception and in line with our previous comment, we confirmed the need of providing a basic training program.
- *Need of additional documentation for the software tools.* The provided software tools were the most valuable aspect of the proposed solution. However, Modeliosoft’s employees remarked the importance of providing a more detailed documentation of such software artifacts so they can be better informed of their usage. This is an important aspect to be addressed in future research cycles.
- *Need of improving the overall explanation of the forecasting solution.* The explanation provided before the execution of the case study to Modeliosoft’s employees was not as effective as expected. The employees’ feedback suggested that instead of being a general explanation of the forecasting solution’s activities, the explanation should particularize on real examples using data from the company. They emphasized the need of showing the results of each step graphically together with real values so that Modeliosoft’s decision-makers (mainly project leaders were mentioned) can better understand the usefulness of the forecasting results and get confidence on them.

9.6 Specifying Learning

Learning and reflection are integrated throughout the action research cycles. Through ongoing reflection during all processes, activities and results, we realized potential improvements as well as consequences that are related to the forecasting solution.

Regarding the **RQ4** related to this thesis, the most important result was the positive perception of Modeliosoft employees regarding the usefulness of the assets and infrastructure from the SESSI method to enable the forecasting of the values of SSIs.

We also gathered rich learning about the forecasting solution itself that will help to improve it:

- *Inclusion of new forecasting techniques*. Although the pilot project did not require other forecasting techniques than the ones included in the proposed solution, we are aware that other projects in Modeliosoft might require the inclusion of other forecasting techniques to get suitable results. The software companion tools were designed with a modular and flexible structure that makes easy to extend them to integrate additional techniques.
- *Inclusion of new performance metrics*. We added a new performance metric called metrics’ contribution error. This metric was not originally included in the forecasting solution, but we realized its potential usefulness for providing insights for improving the SSI Forecasting Model. Other performance metrics may be included in future versions of the forecasting solution.
- *Feedback evaluation results*. The feedback evaluation results obtained from the case study revealed important aspects that might affect the organizational adoption of the forecasting solution. We have assessed them carefully and will serve as an input for planning further cycles of action research.
- *Further support*. One of the specific results from the feedback evaluation performed in the context of the case study, was the need of providing a basic training program to ensure that Modeliosoft’s employees understand and use the forecasting solution in a better way. We have reflected on the steps of the forecasting solution that require better support. We have thought that the initial steps referred to the definition of the forecasting horizon and the expected performance threshold of the SSI Forecasting Model should be further supported. There might be several factors that affect such definitions, but Modeliosoft’s employees are not usually aware of them. For instance, the availability of historical data of the SSI compromises the selection of the forecasting horizon. Or the software development stage that the historical data is pointing out compromises the interpretation of the forecasting results. So, additionally to the basic training program mentioned above, we plan to improve the solution by detailing a set of scenarios together with suggested decisions that maximize the forecasting success.

- *Potential interaction effects between the forecasting results and the decisions taken based on them*. We have thought that there could be some unexpected interaction effects because of the forecasting support in Modeliosoft. For example, if the forecasted values of the SSI show a considerable decrease in some important aspect, it is expected that the company will react to this situation by, for instance, assigning more resources, or in general taking actions to prevent this foresighted situation. These preventive actions will directly impact on the future events related to the SSI and might affect the SSI forecasting itself. This situation was not evidenced in the case study presented in this case study as it did not tackle a longitudinal approach, but we raise this issue as it is of utterly importance to deal with this in future versions of the forecasting solution.

9.7 Limitations and Threads to Validity

Action research has been praised for its utility in practical problem-solving in real world situations but criticized for its transferability and rigor (Ralph et al., 2020). Below we detail the actions performed to strength rigor, trustworthiness, results credibility, and transferability.

- *Smooth collaboration.* The fact that we had previous involvement with Modeliosoft in previous industry-academia collaborations meant an opportunity (instead of a problem), as we were familiar with its context and had built a smooth collaboration environment that undoubtly influenced this project positively. In addition, Modeliosoft's representatives were convinced on their goal of promoting forecasting capabilities and fully supported all related activities of this collaboration. They were always available and willing to provide us any required clarification or feedback for the smooth execution of the whole collaboration.
- *Participant Bias.* Since the beginning of the study, the collaboration team had a clear understanding of the dual objective of the collaboration: improving Modeliosoft's forecasting capabilities by working together to generate and apply useful knowledge for both the company and researchers, which leads to a 'win-win' scenario. To avoid researchers' bias, we conducted the study under the premises of *impartiality,* considering all data gathered and expressed by all participants, and promoting *continuous reflexivity* (individually and as a team)

to make sure that all data is assessed with a clear and unbiased mind. So, we used to share and continuously discuss our impression and notes to be kept at bay pre-existing assumptions.

- *Online interactions:* Given Covid-19 restrictions, most of our interactions in this study were held online instead of face-to-face in the company premises. We are aware that this could have a negative influence on the communication abilities and therefore on our results. However, the fact that we had previous face-to-face meetings with whole team in previous projects (so we knew each other and built in a previous confidence on our commitment, capabilities and trustwothiness) helped to minimize the effects of mostly online interactions.
- *Transferability:* This study was implemented in the specific setting of Modeliosoft and was designed in a way that it is transferable within Modeliosoft (e.g., projects within the organization). This context specific nature of action research hinders the transferability of results to other companies (external validity). However, we would like to emphasize that we tried to provide as much contextual information as possible (while reconciling confidentiality issues) so that other companies can identify their potential similarities and get inspired by the solution proposed in Modeliosoft. In addition, the software tools developed to support the forecasting solution have been released as open source, so others can adapt and use them for similar purposes.

10 Conclusions and Future Work

In this thesis, we have presented a method for supporting and improving the specification, estimation and monitoring of SSI in software development companies and also a demonstration of the usefulness of the resulting assets of the SESSI method for enabling SSI forecasting.

This chapter reviews the main contributions and implications of our research as well as some future lines of investigation which have emerged along our research work. Specifically, Section 10.1 summarizes the main contributions of the thesis, Section 10.2 discusses on the main implications of the thesis on research and practice whilst Section 10.3 relates the envisaged future work.

10.1 Conclusions

The main contributions from this thesis can be summarized into three:

The SESSI method provides support for operationalizing the specification, estimation and monitoring of meaningful SSIs for a software company.

The resulting SSI monitoring infrastructure constructed with the SESSI method proposed in this thesis, is mainly oriented to answer **RQ2** and support the problems related in Chapter 1 (i.e., supporting the specification of company-specific SSIs, dealing with their corresponding estimation’s complexities, and supporting the process of putting forward the mechanisms required to enable their automatic monitoring). Existing proposals, examined to address **RQ1,** that aim to support strategic decision making do not deal with the mentioned problems altogether. Furthermore, existing SSIs detailed in the literature have not been specified with a well-defined method to endorse them, they lack rationale as well as mechanisms to enable automatic estimation and monitoring. All this makes our proposal useful with respect to other proposals dealing with SSIs.

The use of the SESSI method and associated software supporting artifacts have shown promising results to enable an SSI monitoring infrastructure according to the needs and resources of a software company.

The SESSI method proposes a set of interrelated activities using several techniques aimed to deal with the specification, estimation and monitoring of SSIs. The application of the SESSI method in an industrial project, led by **RQ3** has led promising results about their potential applicability and reproducibility: the feedback received scored both aspects as positive, implying that employees of the studied company felt that they would be able to conduct the SESSI method by themselves following the guidelines and software tool support related to the SESSI method.

The SESSI's resulting infrastructure and assets were useful to enable the forecasting of the values of SSIs.

The results of using the resulting models, data and infrastructure from the SESSI method for enabling the forecasting of the values of SSIs were promising (answering **RQ4**). The performed case study shown the suitability and technical feasibility of the forecasting solution.

10.2 Implications

We believe the results presented as part of this thesis might have positive implications for research and practice:

Implications for Research: One the one hand, this thesis analysed and synthetised the current problems and research gaps regarding the use of Data-Driven Decision Making (DDDM) approaches in software companies. Although not directly related to indicators, recent works (Figalist et al., 2021) have also remarked the lack of generic approaches to support decision making processes through the connection of operational data and the information needs at the strategic levels, which is something we have verified in relation to SSIs as shown in Chapter 2. On the other hand, we believe this thesis may advance the state of the art on the use of DDDM approaches in software companies, as we have introduced a generic method to support steps 1-4 of such cycle with the use of SSIs while addressing the unveiled research problems. We believe that researchers can build upon the SESSI method and expand it to cover a wider range of SSIs and even other indicators with similar contextual challenges. Additionally, regarding the forecasting solution for Modeliosoft reported in Chapter 9, we consider it can be useful to visualize how the effort required to conduct the SESSI method may transfer into further decision-making support.

Implications for Practice: The use of the SESSI method and the SSIs derived from its application can help software companies to improve their decision-making processes. SSIs may provide insights of, and into the software development process, allowing companies to identify areas for improvement and make informed decisions regarding resource allocation and project management, among others. This can result in improved software quality, reduced development costs, and increased customer satisfaction. Additionally, the use of the hierarchical definition of an SSIs can also help companies to better communicate with stakeholders by providing diverse views of the factors and metrics that compose an SSI. In addition, regarding the forecasting solution presented in Chapter 9, other companies can identify their potential similarities with the context and requirements of the studied company and get inspired on how to deal with similar forecasting needs. The procedures and software companion tools resulted from this work were released as open source and can be reused and/or adapted by other companies for similar purposes.

10.3 Future Work

The contributions proposed in this thesis open the door to potential future directions, including extensions and improvements.

Regarding improvements to the SESSI method. Based on the feedback and insights obtained during the case study conducted with Modeliosoft in the context of the summative evaluation of the method, new iterations of the method may be considered and conducted. Some of the extensions that can be incorporated into the SESSI method include:

- To include further support and automation to the tool support, in order to ease the probability elicitation tasks (Phase 2).
- To study and consider the SSI estimation models' retraining after their construction and deployment (i.e., after the method's execution) for long-term software projects.

Regarding the cross-applicability of the SESSI method. As the process to derive SSIs through the application of the SESSI method requires the interaction and effort of domain experts from a company, an important aspect to consider is not only to reuse the infrastructure and resulting assets from the SESSI method (as tackled in Chapter 9) but also to reuse the SSI's related assets across different projects of the software company.

The study of the cross-applicability of SSIs can be impacted by a number of factors, including differences in the nature of the software projects, their development methodologies, and the teams involved. So, to ensure the cross-applicability of SSIs we are considering to adjust and refine the SSI specification and estimation model as needed to ensure that they remain relevant and effective across different projects. While it is essential to customize the SSI based on the project requirements and goals, it is important to note that standardizing the software development processes can increase the cross-applicability of SSIs within the same company.

Regarding the applicability of the SESSI method in other contexts. We have considered to apply the method in other contexts not necessarily tied to software development. In this line, it is worth mentioning the participation of the applicant in the DOGO4ML project, in which he has started to explore the applicability of the SESSI method to assess the trustworthiness of AI-Machine Learning systems. In this context, the applicant has developed a software tool able to compute trust-based metrics (i.e., related to performance, fairness, explainability, and robustness) of AI-based models and feed such metrics to the *SSI Estimation* component (described in Chapter 7 and in the Appendix 1 (section 4)) in order to assess the trustworthiness using an estimation model based on Bayesian networks (BNs), likewise the SSI estimation models.

11 References

Antinyan, V., Staron, M., Meding, W., Osterstrom, P., Wikstrom, E., Wranker, J., Henriksson, A., Hansson, J., 2014. Identifying risky areas of software code in Agile/Lean software development: An industrial experience report, in: Proceedings of the 1th Software Evolution Week - IEEE Conference on Software Maintenance, Reengineering, and Reverse Engineering, CSMR-WCRE, '14. IEEE, pp. 154–163. https://doi.org/10.1109/CSMR-WCRE.2014.6747165

Antolić, Ž., 2008. An example of using key performance indicators for software development process efficiency evaluation, in: MIPRO 2008 - 31st International Convention Proceedings: Telecommunications and Information. pp. 156–161.

Aranda, J., Easterbrook, S., 2005. Anchoring and adjustment in software estimation, in: Proceedings of the 10th European Software Engineering Conference Held Jointly with 13th ACM SIGSOFT International Symposium on Foundations of Software Engineering - ESEC/FSE-13. ACM Press, New York, New York, USA, p. 346. https://doi.org/10.1145/1081706.1081761

Armstrong, J.S., 2001. Principles of Forecasting, International Series in Operations Research & Management Science. Springer US, Boston, MA. https://doi.org/10.1007/978-0-306-47630-3

Assimakopoulos, V., Nikolopoulos, K., 2000. The theta model: a decomposition approach to forecasting. Int. J. Forecast. 16, 521–530. https://doi.org/10.1016/S0169-2070(00)00066-2

Asthana, A., Olivieri, J., 2009. Quantifying software reliability and readiness, in: Proceedings of the IEEE International Workshop Technical Committee on Communications Quality and Reliability, CQR '09. IEEE, pp. 1–6. https://doi.org/10.1109/CQR.2009.5137352

Aurum, A., Wohlin, C., Porter, A., 2006. Aligning software project decisions: A case study. Int. J. Softw. Eng. Knowl. Eng. 16, 795–818. https://doi.org/10.1142/S0218194006003002

Avison, D., 2003. Action research: a research approach for cooperative work, in: Proceedings of the 7th International Conference on Computer Supported Cooperative Work in Design, CSCWD ’03. COPPE/UFRJ, pp. 19–24. https://doi.org/10.1109/CSCWD.2002.1047641

Avison, D.E., Lau, F., Myers, M.D., Nielsen, P.A., 1999. Action research. Commun. ACM 42, 94–97. https://doi.org/10.1145/291469.291479

Bakota, T., Hegedus, P., Ladanyi, G., Kortvelyesi, P., Ferenc, R., Gyimothy, T., 2012. A cost model based on software maintainability. IEEE Int. Conf. Softw. Maintenance, ICSM 316–325. https://doi.org/10.1109/ICSM.2012.6405288

Barone, D., Jiang, L., Amyot, D., Mylopoulos, J., 2011. Composite Indicators for Business Intelligence, in: Conceptual Modeling – ER 2011. 30th International Conference on Conceptual Modeling. pp. 448–458. https://doi.org/10.1007/978-3-642-24606-7_35

Barreto, A.O.S., Rocha, A.R., 2010. Analyzing the Similarity among Software Projects to Improve Software Project Monitoring Processes, in: 2010 Seventh International Conference on the Quality of Information and Communications Technology. IEEE, pp. 441–446. https://doi.org/10.1109/QUATIC.2010.79

Basili, V., Lindvall, M., Regardie, M., Seaman, C., Heidrich, J., Münch, J., Rombach, D., Trendowicz, A., 2007. Bridging the gap between business strategy and software development, in: ICIS 2007 Proceedings - Twenty Eighth International Conference on Information Systems.

Basili, V.R., Lindvall, M., Regardie, M., Seaman, C., Heidrich, J., Münch, J., Rombach, D., Trendowicz, A., 2010. Linking Software Development and Business Strategy Through Measurement. Computer (Long. Beach. Calif). 43, 57–65. https://doi.org/10.1109/MC.2010.108

Bastarrica, M.C., Perovich, D., Marín, J., Rioseco, L., María, C., Bastarrica, D., Perovich, J., Marín, L.R., 2017. Process-Based Project Management and SPI. Proc. 2017 Int. Conf. Softw. Syst. Process 10. https://doi.org/10.1145/3084100

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J., Thomas, D., 2001. Manifesto for Agile Software Development. Manif. Agil. Softw. Dev. URL http://www.agilemanifesto.org/

Béland, S., Abran, A., 2012. A measurement framework to support continuous improvement in software intensive organizations. Proc. 2012 Jt. Conf. 22nd Int. Work. Softw. Meas. 2012 7th Int. Conf. Softw. Process Prod. Meas. IWSM-MENSURA 2012 215–220. https://doi.org/10.1109/IWSM-MENSURA.2012.38

Bergmeir, C., Hyndman, R.J., Benítez, J.M., 2016. Bagging exponential smoothing methods using STL decomposition and Box–Cox transformation. Int. J. Forecast. 32, 303–312. https://doi.org/10.1016/J.IJFORECAST.2015.07.002

Bezerra, C.I.M., Coelho, C.C., Pires, C.G.S., Albuquerque, A.B., 2010. A practical application of performance models to predict the productivity of projects. Innov. Adv. Comput. Sci. Eng. 273–277. https://doi.org/10.1007/978-90-481-3658-2_47/COVER

Biddle, R., Meier, A., Kropp, M., Anslow, C., 2018. Myagile, in: Proceedings of the 11th International Workshop on Cooperative and Human Aspects of Software Engineering. ACM, New York, NY, USA, pp. 73–76. https://doi.org/10.1145/3195836.3195845

Boldt, M., Jacobsson, A., Baca, D., Carlsson, B., 2017. Introducing a Novel Security-Enhanced Agile Software Development Process. Int. J. Secur. Softw. Eng. 8, 26–52. https://doi.org/10.4018/IJSSE.2017040102:

Budgen, D., Turner, M., Brereton, P., Kitchenham, B., 2008. Using Mapping Studies in Software Engineering, in: PPIG 2008: 20th Annual Meeting of the Psychology of Programming Interest Group. pp. 195–204.

Buenen, M., Walgude, A., 2018. World Quality Report 2018–19, Capgemini, Sogeti and Micro Focus. URL https://www.sogeti.com/explore/reports/world-quality-report-201819/

Çalıklı Chalmers, G., Meding, W., 2018. Measure Early and Decide Fast: Transforming Quality Management and Measurement to Continuous Deployment, in: Proceedings of the 2018 International Conference on Software and System Process. ACM, New York, NY, USA, pp. 51–60. https://doi.org/10.1145/3202710

Carvallo, J.P., Franch, X., Grau, G., Quer, C., 2004. COSTUME: A method for building quality models for composite COTS-based software systems, in: Proceedings of the 4th International Conference on Quality Software, QSIC '04. IEEE, pp. 214–221. https://doi.org/10.1109/QSIC.2004.1357963

Castro, O., Espinoza, A., Martínez-Martínez, A., 2012. Estimating the software product value during the development process, in: International Conference on Product Focused Software Process Improvement. Springer, Berlin, Heidelberg, pp. 74–88. https://doi.org/10.1007/978-3-642-31063-8_7/COVER

Cerqueira, V., Torgo, L., Mozetič, I., 2020. Evaluating time series forecasting models: an empirical study on performance estimation methods. Mach. Learn. 109, 1997–2028. https://doi.org/10.1007/s10994-020-05910-7

Chambers, J.C., Mullick, S.K., Smith, D.D., 1971. How to Choose the Right Forecasting Technique [WWW Document]. URL https://hbr.org/1971/07/how-to-choose-the-right-forecasting-technique (accessed 4.29.21).

Chang, C.P., 2015. Software Risk Modeling by Clustering Project Metrics. Int. J. Softw. Eng. Knowl. Eng. 25, 1053–1076. https://doi.org/10.1142/S0218194015500175

Chen, N., Hoi, S.C.H., Xiao, X., 2014. Software process evaluation: a machine learning framework with application to defect management process. Empir. Softw. Eng. 19, 1531–1564. https://doi.org/10.1007/S10664-013-9254-Z/TABLES/12

Chen, N., Hoi, S.C.H., Xiao, X., 2011. Software process evaluation: A machine learning approach. 2011 26th IEEE/ACM Int. Conf. Autom. Softw. Eng. ASE 2011, Proc. 333–342. https://doi.org/10.1109/ASE.2011.6100070

Chen, Y.C., Wheeler, T.A., Kochenderfer, M.J., 2017. Learning Discrete Bayesian Networks from Continuous Data. J. Artif. Intell. Res. 59, 103–132. https://doi.org/10.1613/JAIR.5371

Cito, J., 2016. Developer targeted analytics: Supporting software development decisions with runtime information. ASE 2016 - Proc. 31st IEEE/ACM Int. Conf. Autom. Softw. Eng. 892–895. https://doi.org/10.1145/2970276.2975939

Cleveland, R.B., Cleveland, W.S., McRae, J.E., Terpenning, I., 1990. STL: A Seasonal-Trend Decomposition Procedure Based on Loess (with Discussion). J. Off. Stat. 6, 3–73.

Colombo, A., Damiani, E., Frati, F., Oltolina, S., Reed, K., Ruffatti, G., 2008. The use of a meta-model to support multi-project process measurement. Neonatal, Paediatr. Child Heal. Nurs. 503–510. https://doi.org/10.1109/APSEC.2008.55

Constantinou, A., Fenton, N., 2017. Towards smart-data: Improving predictive accuracy in long-term football team performance. Knowledge-Based Syst. https://doi.org/10.1016/j.knosys.2017.03.005

Cuadrado-García, J.L., Cuadrado-Gallego, J.J., Herranz-Martínez, M.A., Rodríguez-Soria, P., 2011. Improve tracking in the software development projects. Proc. - Jt. Conf. 21st Int. Work. Softw. Meas. IWSM 2011 6th Int. Conf. Softw. Process Prod. Meas. MENSURA 2011 215–220. https://doi.org/10.1109/IWSM-MENSURA.2011.10

Dam, H.K., Tran, T., Ghose, A., 2018. Explainable software analytics. Proc. - Int. Conf. Softw. Eng. 53–56. https://doi.org/10.1145/3183399.3183424

Das, B., 2004. Generating Conditional Probabilities for Bayesian Networks: Easing the Knowledge Acquisition Problem. CoRR 1–24. https://doi.org/DSTO-TR-0918

De Aquino Júnior, G.S., De Lemos Meira, S.R., 2009. Towards effective productivity measurement in software projects. 4th Int. Conf. Softw. Eng. Adv. ICSEA 2009, Incl. SEDES 2009 Simp. para Estud. Doutor. em Eng. Softw. 241–249. https://doi.org/10.1109/ICSEA.2009.44

De Livera, A.M., Hyndman, R.J., Snyder, R.D., 2011. Forecasting Time Series With Complex Seasonal Patterns Using Exponential Smoothing. J. Am. Stat. Assoc. 106, 1513–1527. https://doi.org/10.1198/jasa.2011.tm09771

Development Core Team, R., 2008. R: A Language and Environment for Statistical Computing. Vienna Austria R Found. Stat. Comput.

Díaz-Ley, M., García, F., Piattini, M., 2008. Implementing a software measurement program in small and medium enterprises: A suitable framework. IET Softw. 2, 417–436. https://doi.org/10.1049/IET-SEN:20080026

Digital.ai, 2021. 15th State of Agile Report. URL https://digital.ai/resource-center/analyst-reports/state-of-agile-report

Dikici, A., Turetken, O., Demirors, O., 2012. A case study on measuring process quality: Lessons learned. Proc. - 38th EUROMICRO Conf. Softw. Eng. Adv. Appl. SEAA 2012 294–297. https://doi.org/10.1109/SEAA.2012.26

Dougherty, J., Kohavi, R., Sahami, M., 1995. Supervised and Unsupervised Discretization of Continuous Features, in: Machine Learning Proceedings 1995. Elsevier, pp. 194–202. https://doi.org/10.1016/b978-1-55860-377-6.50032-3

Edison, H., Wang, X., Conboy, K., 2022. Comparing Methods for Large-Scale Agile Software Development: A Systematic Literature Review. IEEE Trans. Softw. Eng. 48, 2709–2731. https://doi.org/10.1109/TSE.2021.3069039

Elden, M., Chisholm, R.F., 1993. Emerging Varieties of Action Research: Introduction to the Special Issue. Hum. Relations 46, 121–142. https://doi.org/10.1177/001872679304600201

Endres, D.M., Schindelin, J.E., 2003. A new metric for probability distributions. IEEE Trans. Inf. Theory 49, 1858–1860. https://doi.org/10.1109/TIT.2003.813506

Ericsson, W.M., Sweden, G., 2017. Effective Monitoring of Progress of Agile Software Development Teams in Modern Software Companies - An Industrial Case Study. Proc. 27th Int. Work. Softw. Meas. 12th Int. Conf. Softw. Process Prod. Meas. 11, 23–32. https://doi.org/10.1145/3143434

Fenton, N., Neil, M., 2005. Ranked nodes : A simple and effective way to model qualitative judgements in large-scale. URL http://qmro.qmul.ac.uk/xmlui/handle/123456789/5046 (accessed 5.17.22).

Fenton, N.E., Neil, M., Caballero, J.G., 2007. Using Ranked Nodes to Model Qualitative Judgments in Bayesian Networks. IEEE Trans. Knowl. Data Eng. 19, 1420–1432. https://doi.org/10.1109/TKDE.2007.1073

Figalist, I., Elsner, C., Bosch, J., Holmstrom Olsson, H., 2019. Business as Unusual: A Model for Continuous Real-Time Business Insights Based on Low Level Metrics. Proc. - 45th Euromicro Conf. Softw. Eng. Adv. Appl. SEAA 2019 66–73. https://doi.org/10.1109/SEAA.2019.00019

Figalist, I., Elsner, C., Bosch, J., Olsson, H.H., 2022. Breaking the vicious circle: A case study on why AI for software analytics and business intelligence does not take off in practice. J. Syst. Softw. 184, 111135. https://doi.org/10.1016/j.jss.2021.111135

Figalist, I., Elsner, C., Bosch, J., Olsson, H.H., 2021. Fast and curious: A model for building efficient monitoring- and decision-making frameworks based on quantitative data. Inf. Softw. Technol. 132, 106458. https://doi.org/10.1016/j.infsof.2020.106458

Franch, X., Ayala, C., Lopez, L., Martinez-Fernandez, S., Rodriguez, P., Gomez, C., Jedlitschka, A., Oivo, M., Partanen, J., Raty, T., Rytivaara, V., 2017. Data-Driven Requirements Engineering in Agile Projects: The Q-Rapids Approach, in: Proceedings of the IEEE 25th International Requirements Engineering Conference Workshops, REW ’17. IEEE, pp. 411–414. https://doi.org/10.1109/REW.2017.85

Freire, A., Perkusich, M., Saraiva, R., Almeida, H., Perkusich, A., 2018. A Bayesian networks-based approach to assess and improve the teamwork quality of agile teams. Inf. Softw. Technol. 100, 119–132. https://doi.org/10.1016/j.infsof.2018.04.004

Galdi, P., Tagliaferri, R., 2018. Data mining: Accuracy and error measures for classification and prediction, in: Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics. https://doi.org/10.1016/B978-0-12-809633-8.20474-3

García, F., Piattini, M., Ruiz, F., Canfora, G., Visaggio, C.A., 2006. FMESP: Framework for the modeling and evaluation of software processes. J. Syst. Archit. 52, 627–639. https://doi.org/10.1016/J.SYSARC.2006.06.007

García, F., Ruiz, F., Cruz, J.A., Piattini, M., 2003. Integrated measurement for the evaluation and improvement of software processes, in: European Workshop on Software Process Technology. Springer Verlag, pp. 94–111. https://doi.org/10.1007/978-3-540-45189-1_8

García, F., Serrano, M., Cruz-Lemus, J., Ruiz, F., Piattini, M., 2007. Managing software process measurement: A metamodel-based approach. Inf. Sci. (Ny). 177, 2570–2586. https://doi.org/10.1016/J.INS.2007.01.018

Gill, B., Borden, B., Hallgren, K., 2014. A Conceptual Framework for Data-Driven Decision Making. Princeton. URL https://www.mathematica.org/our-publications-and-findings/publications/a-conceptual-framework-for-data-driven-decision-making (accessed 4.11.21).

Goepel, K.D., 2018. Implementation of an Online Software Tool for the Analytic Hierarchy Process (AHP-OS). Int. J. Anal. Hierarchy Process 10. https://doi.org/10.13033/ijahp.v10i3.590

Green, K.C., Armstrong, J.S., 2015. Simple versus complex forecasting: The evidence. J. Bus. Res. 68, 1678–1685. https://doi.org/10.1016/j.jbusres.2015.03.026

Gren, L., Svensson, R.B., Unterkalmsteiner, M., 2017. Is It Possible to Disregard Obsolete Requirements? An Initial Experiment on a Potentially New Bias in Software Effort Estimation, in: 2017 IEEE/ACM 10th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE). IEEE, pp. 56–61. https://doi.org/10.1109/CHASE.2017.10

Guceglioglu, A.S., Demirors, O., 2005. A process based model for measuring process quality attributes, in: European Conference on Software Process Improvement. Springer, Berlin, Heidelberg, pp. 118–129. https://doi.org/10.1007/11586012_12

Halford, M., 2022. sorobn: Bayesian networks in Python. URL https://github.com/MaxHalford/sorobn (accessed 2.9.23).

Hao, W., Haiqing, W., Hefei, Z., 2008. Software productivity analysis with CSBSG data set. Proc. - Int. Conf. Comput. Sci. Softw. Eng. CSSE 2008 2, 587–593. https://doi.org/10.1109/CSSE.2008.1178

Hearty, P., Fenton, N., Marquez, D., Neil, M., 2009. Predicting project velocity in XP using a learning dynamic Bayesian network model. IEEE Trans. Softw. Eng. 35, 124–137. https://doi.org/10.1109/TSE.2008.76

Hyndman, R., Koehler, A., Ord, K., Snyder, R., 2008. Forecasting with Exponential Smoothing, Springer Series in Statistics, Springer Series in Statistics. Springer Berlin Heidelberg, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-71918-2

Hyndman, R., Kostenko, A., 2007. Minimum sample size requirements for seasonal forecasting models. Foresight Int. J. Appl. Forecast. 6, 12–15.

Hyndman, R.J., Athanasopoulos, G., 2018. Forecasting: Principles and Practice, 2nd editio. ed. OTexts, Melbourne, Australia.

Hyndman, R.J., Khandakar, Y., 2008. Automatic Time Series Forecasting: The forecast Package for R. J. Stat. Softw. 27. https://doi.org/10.18637/jss.v027.i03

Hyndman, R.J., Koehler, A.B., 2006. Another look at measures of forecast accuracy. Int. J. Forecast. 22, 679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001

IEEE, 2017. 15939-2017 - ISO/IEC/IEEE International Standard - Systems and software engineering--Measurement process, IEEE. IEEE. https://doi.org/10.1109/IEEESTD.2017.7907158

Ikemoto, S., Dohi, T., Okamura, H., 2013. Estimating software reliability with static project data in incremental development processes. Proc. - Jt. Conf. 23rd Int. Work. Softw. Meas. 8th Int. Conf. Softw. Process Prod. Meas. IWSM-MENSURA 2013 219–224. https://doi.org/10.1109/IWSM-MENSURA.2013.38

Ilieva, S., Ivanov, P., Stefanova, E., 2004. Analyses of an agile methodology implementation, in: Proceedings of the 30th Euromicro Conference, EUROMICRO ’04. IEEE, pp. 326–333. https://doi.org/10.1109/EURMIC.2004.1333387

Janssen, M., van der Voort, H., Wahyudi, A., 2017. Factors influencing big data decision-making quality. J. Bus. Res. 70, 338–345. https://doi.org/10.1016/j.jbusres.2016.08.007

Johnson, P., Lagerström, R., Närman, P., Simonsson, M., 2007. Enterprise architecture analysis with extended influence diagrams. Inf. Syst. Front. 9, 163–180. https://doi.org/10.1007/s10796-007-9030-y

Johnson, P.M., Kou, H., Paulding, M., Zhang, Q., Kagawa, A., Yamashita, T., 2005. Improving software development management through software project telemetry. IEEE Softw. 22, 76–85. https://doi.org/10.1109/MS.2005.95

Jørgensen, M., Sjøberg, D.I.K., 2004. The impact of customer expectation on software development effort estimates. Int. J. Proj. Manag. 22, 317–325. https://doi.org/10.1016/S0263-7863(03)00085-1

Keser, B., Iyidogan, T., Ozkan, B., 2013. ASSIST: An integrated measurement tool. Proc. - Jt. Conf. 23rd Int. Work. Softw. Meas. 8th Int. Conf. Softw. Process Prod. Meas. IWSM-MENSURA 2013 237–242. https://doi.org/10.1109/IWSM-MENSURA.2013.41

Khokhar, M. Nawazish, Rehman, S.U., Mansoor, A., Khokhar, Muhammad Nadeem, Rauf, A., 2010. MECA: Software process improvement for small organizations. 2010 Int. Conf. Inf. Emerg. Technol. ICIET 2010. https://doi.org/10.1109/ICIET.2010.5625678

Kincaid, D., Cheney, W., 2002. Numerical Analysis: Mathematics of Scientific Computing, 3rd Revise. ed. American Mathematical Society.

Kitchenham, B., Charters, S.M., 2007. Guidelines for performing Systematic Literature Reviews in Software Engineering, EBSE Technical Report. EBSE-2007-01.

Kojima, T., Hasegawa, T., Misumi, M., Nakamura, T., 2008. Risk analysis of software process measurements. Softw. Qual. J. 16, 361–376. https://doi.org/10.1007/S11219-007-9040-5/FIGURES/7

Krippendorff, K., 1980. Content analysis: an introduction to its methodology. SAGE Publications.

Kumar, C., Yadav, D.K., 2015. A Probabilistic Software Risk Assessment and Estimation Model for Software Projects. Procedia Comput. Sci. 54, 353–361. https://doi.org/10.1016/j.procs.2015.06.041

Lami, G., Fabbrini, F., Fusani, M., 2013. A methodology to derive sustainability indicators for software development projects, in: Proceedings of the 2013 International Conference on Software and System Process. ACM, New York, NY, USA, pp. 70–77. https://doi.org/10.1145/2486046.2486060

Layman, L., Williams, L., Cunningham, L., 2004. Exploring Extreme Programming in Context: An Industrial Case Study, in: Agile Development Conference. IEEE, pp. 32–41. https://doi.org/10.1109/ADEVC.2004.15

Lethbridge, T.C., Sim, S.E., Singer, J., 2005. Studying Software Engineers: Data Collection Techniques for Software Field Studies. Empir. Softw. Eng. 2005 103 10, 311–341. https://doi.org/10.1007/S10664-005-1290-X

Lin, C., Huang, Z., 2009. A flexible metric-driven framework for software process. NCM 2009 - 5th Int. Jt. Conf. INC, IMS, IDC 1198–1202. https://doi.org/10.1109/NCM.2009.189

Lin, J., 1991. Divergence Measures Based on the Shannon Entropy. IEEE Trans. Inf.

Theory 37, 145–151. https://doi.org/10.1109/18.61115

List, B., Bruckner, R.M., Kapaun, J., 2005. Holistic software process performance measurement from the stakeholders' perspective. Proc. - Int. Work. Database Expert Syst. Appl. DEXA 2006, 941–947. https://doi.org/10.1109/DEXA.2005.109

López, L., Burgués, X., Martínez-Fernández, S., Vollmer, A.M., Behutiye, W., Karhapää, P., Franch, X., Rodríguez, P., Oivo, M., 2022. Quality measurement in agile and rapid software development: A systematic mapping. J. Syst. Softw. 186, 111187. https://doi.org/10.1016/J.JSS.2021.111187

López, L., Manzano, M., Gómez, C., Oriol, M., Farré, C., Franch, X., Martínez-Fernández, S., Vollmer, A.M., 2021. QaSD: A Quality-aware Strategic Dashboard for supporting decision makers in Agile Software Development. Sci. Comput. Program. 202, 102568. https://doi.org/10.1016/j.scico.2020.102568

Mahnič, V., Vrana, I., 2007. Using stakeholder-driven process performance measurement for monitoring the performance of a Scrum-based software development process. Elektroteh. Vestn. 74, 241–247.

Mahnic, V., Zabkar, N., 2008. Measurement repository for Scrum-based software development process, in: Proceedings of the 2nd WSEAS International Conference on Computer Engineering and Applications. pp. 23–28.

Makridakis, S., Hibon, M., 2000. The M3-Competition: results, conclusions and implications. Int. J. Forecast. 16, 451–476. https://doi.org/10.1016/S0169-2070(00)00057-1

Makridakis, S., Spiliotis, E., Assimakopoulos, V., 2018. Statistical and Machine Learning forecasting methods: Concerns and ways forward. PLoS One 13, e0194889. https://doi.org/10.1371/JOURNAL.PONE.0194889

Mandinach, E., Honey, M., Light, D., 2006. A Theoretical Framework for Data-Driven Decision Making. URL https://cct.edc.org/sites/cct.edc.org/files/publications/DataFrame_AERA06.pdf (accessed 11.16.22).

Mann, C., Maurer, F., 2005. A case study on the impact of scrum on overtime and customer satisfaction, in: Agile Development Conference (ADC'05). IEEE

Comput. Soc, pp. 70–79. https://doi.org/10.1109/ADC.2005.1

Manzano, M., 2022a. Process Measurement SMS - Papers selection process. URL https://docs.google.com/spreadsheets/d/1Y3CcrVttcQqWyTEOrZbfcURh2X8Zm_dX

Manzano, M., 2022b. Quality Measurement SMS - Papers selection process. URL https://drive.google.com/file/d/1meDeZ6bLmEGY_1SsbB5UCTwvlfiSxBYL

Manzano, M., 2022c. Model Comparison Tool. URL https://github.com/martimanzano/SSI-forecast/blob/IST-paper/src/main/java/Forecast/Utils.java#L190

Manzano, M., 2022d. Accuracy Computation Tool. URL https://github.com/martimanzano/SSI-forecast/blob/IST-paper/src/main/java/Forecast/Utils.java#L359

Manzano, M., 2022e. Forecasting Report Tool. URL https://github.com/martimanzano/SSI-forecast/blob/IST-paper/src/main/java/Forecast/Utils.java#L395

Manzano, M., 2022f. Indicator Forecasting Computation. URL https://github.com/martimanzano/SSI-forecast/tree/IST-paper (Java). https://github.com/martimanzano/SSI-forecast-R_scripts/tree/IST-paper (R)

Manzano, M., 2021a. Time Series Gathering Tool. URL https://github.com/martimanzano/SSI-forecast-R_scripts/blob/IST-paper/TimeSeriesFunctions_GPL_R_elastic_1.R#L25

Manzano, M., 2021b. Autocorrelation Test Tool. URL https://github.com/martimanzano/SSI-forecast-R_scripts/blob/IST-paper/TimeSeriesFunctions_GPL_R_elastic_1.R#L539 (R). https://github.com/martimanzano/SSI-forecast/blob/IST-paper/src/main/java/Forecast/Elastic_RForecast.java#L279 (Java)

Manzano, M., 2021c. Model Fitting Tool. URL https://github.com/martimanzano/SSI-forecast-R_scripts/blob/IST-paper/TimeSeriesFunctions_GPL_R_elastic_1.R#L159 (R). https://github.com/martimanzano/SSI-forecast/blob/IST-paper/src/main/java/Forecast/Elastic_RForecast.java#L410 (Java)

Manzano, M., 2021d. Forecasting Execution Tool. URL https://github.com/martimanzano/SSI-forecast-R_scripts/blob/IST-paper/TimeSeriesFunctions_GPL_R_elastic_1.R#L173 (R). https://github.com/martimanzano/SSI-forecast/blob/IST-paper/src/main/java/Forecast/Elastic_RForecast.java#L480 (Java)

Manzano, M., 2021e. Distance Computation Tool. URL https://github.com/martimanzano/SSI-assessment/blob/3d877924310473cf172959ac56efe352f8255c56/src/main/java/Util_Assessment_SI/BayesUtils.java#L171

Manzano, M., Ayala, C., Gómez, C., Abherve, A., Franch, X., Mendes, E., 2021. A Method to Estimate Software Strategic Indicators in Software Development: An Industrial Application. Inf. Softw. Technol. 129, 106433. https://doi.org/10.1016/j.infsof.2020.106433

Manzano, M., Ayala, C., Gomez, C., Lopez Cuesta, L., 2019. A Software Service Supporting Software Quality Forecasting, in: 2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C). IEEE, pp. 130–132. https://doi.org/10.1109/QRS-C.2019.00037

Manzano, M., Gomez, C., Ayala, C., Martinez-Fernandez, S., Ram, P., Rodriguez, P., Oriol, M., 2018a. Definition of the On-time Delivery Indicator in Rapid Software Development, in: Proceedings of the IEEE 1st International Workshop on Quality Requirements in Agile Projects, QuaRAP '18. IEEE, pp. 1–5. https://doi.org/10.1109/QuaRAP.2018.00006

Manzano, M., Mendes, E., Gómez, C., Ayala, C., Franch, X., 2018b. Using Bayesian Networks to estimate Strategic Indicators in the context of Rapid Software Development, in: Proceedings of the 14th International Conference on Predictive Models and Data Analytics in Software Engineering, PROMISE '18. ACM, New York, NY, USA, pp. 52–55. https://doi.org/10.1145/3273934.3273940

Martinez-Fernandez, S., Jedlitschka, A., Guzman, L., Vollmer, A.M., 2018. A Quality Model for Actionable Analytics in Rapid Software Development, in: Proceedings of the 44th Euromicro Conference on Software Engineering and Advanced

Applications, SEAA '18. IEEE, pp. 370–377. https://doi.org/10.1109/SEAA.2018.00067

Martínez-Fernández, S., Jovanovic, P., Franch, X., Jedlitschka, A., 2018. Towards automated data integration in software analytics, in: BIRTE '18: Proceedings of the International Workshop on Real-Time Business Intelligence and Analytics. Association for Computing Machinery, pp. 1–5. https://doi.org/10.1145/3242153.3242159

Martinez-Fernandez, S., Vollmer, A.M., Jedlitschka, A., Franch, X., Lopez, L., Ram, P., Rodriguez, P., Aaramaa, S., Bagnato, A., Choras, M., Partanen, J., 2019. Continuously Assessing and Improving Software Quality With Software Analytics Tools: A Case Study. IEEE Access 7, 68219–68239. https://doi.org/10.1109/ACCESS.2019.2917403

Matthies, C., Hesse, G., 2019. Towards using Data to Inform Decisions in Agile Software Development: Views of Available Data, in: Proceedings of the 14th International Conference on Software Technologies, ICSOFT '19. SCITEPRESS - Science and Technology Publications, pp. 552–559. https://doi.org/10.5220/0007967905520559

Matthies, C., Kowark, T., Richly, K., Uflacker, M., Plattner, H., 2016. ScrumLint: Identifying Violations of Agile Practices Using Development Artifacts. Proc. 9th Int. Work. Coop. Hum. Asp. Softw. Eng. https://doi.org/10.1145/2897586

Meidan, A., García-García, J.A., Ramos, I., Escalona, M.J., 2018. Measuring Software Process. ACM Comput. Surv. 51, 1–32. https://doi.org/10.1145/3186888

Mendes, E., 2014. Practitioner's knowledge representation: A pathway to improve software effort estimation, Practitioner's Knowledge Representation: A Pathway to Improve Software Effort Estimation. Springer Berlin Heidelberg, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-54157-5

Mendes, E., Rodriguez, P., Freitas, V., Baker, S., Atoui, M.A., 2018. Towards improving decision making and estimating the value of decisions in value-based software engineering: the VALUE framework. Softw. Qual. J. 26, 607–656. https://doi.org/10.1007/s11219-017-9360-z

Mesquida Calafat, A.L., Mas, A., Pacheco, M., 2022. Fake Agile: What Is It and How to Avoid It? IT Prof. 24, 69–73. https://doi.org/10.1109/MITP.2021.3139826

Mikkonen, T., Lassenius, C., Männistö, T., Oivo, M., Järvinen, J., 2018. Continuous and collaborative technology transfer: Software engineering research with real-time industry impact. Inf. Softw. Technol. 95, 34–45. https://doi.org/10.1016/J.INFSOF.2017.10.013

Moe, N.B., Aurum, A., Dybå, T., 2012. Challenges of shared decision-making: A multiple case study of agile software development. Inf. Softw. Technol. 54, 853–865. https://doi.org/10.1016/j.infsof.2011.11.006

Monteiro, L.F.S., De Oliveira, K.M., 2011. Defining a catalog of indicators to support process performance analysis. J. Softw. Maint. Evol. Res. Pract. 23, 395–422. https://doi.org/10.1002/SMR.482

Newbold, P., 1983. ARIMA model building and the time series analysis approach to forecasting. J. Forecast. 2, 23–35. https://doi.org/10.1002/for.3980020104

Oates, B.J., 2006. Researching Information Systems and Computing. Sage Publications Ltd.

Olague, H.M., Etzkorn, L.H., Li, W., Cox, G., 2006. Assessing design instability in iterative (agile) object-oriented projects. J. Softw. Maint. Evol. Res. Pract. 18, 237–266. https://doi.org/10.1002/SMR.332

Olszak, C.M., 2016. Toward Better Understanding and Use of Business Intelligence in Organizations. Inf. Syst. Manag. 33, 105–123. https://doi.org/10.1080/10580530.2016.1155946

Padmini, K.V.J., Dilum Bandara, H.M.N., Perera, I., 2015. Use of software metrics in agile software development process, in: 2015 Moratuwa Engineering Research Conference (MERCon). IEEE, pp. 312–317. https://doi.org/10.1109/MERCon.2015.7112365

Pérez Torres, A., Gómez Seoane, C., Manzano Aguilar, M., 2021. Desenvolupament d'un sistema software per a la creació d'Indicadors Estratègics (SI) utilitzant Xarxes Bayesianes. Universitat Politècnica de Catalunya. URL https://upcommons.upc.edu/handle/2117/343847 (accessed 12.9.21).

Perkusich, M., Gorgônio, K.C., Almeida, H., Perkusich, A., 2017. Assisting the continuous improvement of Scrum projects using metrics and Bayesian networks. J. Softw. Evol. Process 29, e1835. https://doi.org/10.1002/SMR.1835

Perkusich, M., Soares, G., Almeida, H., Perkusich, A., 2015. A procedure to detect problems of processes in software development projects using Bayesian networks. Expert Syst. Appl. 42, 437–450. https://doi.org/10.1016/j.eswa.2014.08.015

Petersen, K., Wohlin, C., 2011. Measuring the flow in lean software development. Softw. Pract. Exp. 41, 975–996. https://doi.org/10.1002/spe.975

Q-Rapids, 2019a. Q-Rapids Deliverable D3.4. URL https://ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5c64f4619&appId=PPGMS (accessed 12.9.21).

Q-Rapids, 2019b. Q-Rapids Deliverable D3.5. URL https://ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5c8d86f40&appId=PPGMS

Q-Rapids, 2019c. Quality-aware Rapid Software development project. European Union’s Horizon 2020 research and innovation programme under grant agreement No 732253. URL https://www.q-rapids.eu (accessed 6.13.20).

Q-Rapids, 2018a. Q-Rapids Deliverable D3.1. URL https://www.q-rapids.eu/deliverables

Q-Rapids, 2018b. Q-Rapids Deliverable D1.1. URL https://www.q-rapids.eu/deliverables

Q-Rapids, 2017. Q-Rapids Deliverable D5.1. URL https://www.q-rapids.eu/deliverables

Quah, J.T.S., Liew, S.W., 2008. Gauzing software readiness using metrics. SMCia/08 - Proc. 2008 IEEE Conf. Soft Comput. Ind. Appl. 426–431. https://doi.org/10.1109/SMCIA.2008.5046002

Ralph, P., Baltes, S., Bianculli, D., Dittrich, Y., Felderer, M., Feldt, R., Filieri, A., Furia, C.A., Graziotin, D., He, P., Hoda, R., Juristo, N., Kitchenham, B., Robbes, R., Mendez, D., Molleri, J., Spinellis, D., Staron, M., Stol, K., Tamburri, D., Torchiano, M., Treude, C., Turhan, B., Vegas, S., 2020. ACM SIGSOFT Empirical

Standards.

Raschka, S., 2018. Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning, arXiv. URL http://arxiv.org/abs/1811.12808 (accessed 1.24.20).

Roden, P.L., Virani, S., Etzkorn, L.H., Messimer, S., 2007. An Empirical Study of the Relationship of Stability Metrics and the QMOOD Quality Models Over Software Developed Using Highly Iterative or Agile Software Processes, in: Seventh IEEE International Working Conference on Source Code Analysis and Manipulation (SCAM 2007). IEEE, pp. 171–179. https://doi.org/10.1109/SCAM.2007.29

Runeson, P., Höst, M., 2009. Guidelines for conducting and reporting case study research in software engineering. Empir. Softw. Eng. https://doi.org/10.1007/s10664-008-9102-8

Runeson, P., Höst, M., Rainer, A., Regnell, B., 2012. Case Study Research in Software Engineering, Case Study Research in Software Engineering: Guidelines and Examples. John Wiley & Sons, Inc., Hoboken, NJ, USA. https://doi.org/10.1002/9781118181034

Salo, O., Tihinen, M., Vierimaa, M., 2002. Enabling comprehensive use of metrics, in: International Conference on Product Focused Software Process Improvement. Springer Verlag, pp. 326–336. https://doi.org/10.1007/3-540-36209-6_28

Schackmann, H., Jansen, M., Lischkowitz, C., Lichter, H., 2009. QMetric - A metric tool suite for the evaluation of software process data. 2009 31st Int. Conf. Softw. Eng. - Companion Vol. ICSE 2009 415–416. https://doi.org/10.1109/ICSE-COMPANION.2009.5071039

Schelter, S., Biessmann, F., Januschowski, T., Salinas, D., Seufert, S., Szarvas, G., 2018. On Challenges in Machine Learning Model Management. IEEE Data Eng. Bull. 41, 5–15.

Shahnewaz, S.M., Ruhe, G., 2014. RELREA-An Analytical Approach for Evaluating Release Readiness, in: 26th International Conference on Software Engineering and Knowledge Engineering 2014. pp. 437–442.

Shashi, K.N.R., Nair, T.R.G., Suma, V., 2014. SLI, a new metric to determine success of a software project, in: 2014 International Conference on Electronics and

Communication Systems (ICECS). IEEE, pp. 1–5. https://doi.org/10.1109/ECS.2014.6892814

Shaub, D., Ellis, P., 2020. forecastHybrid: Convenient Functions for Ensemble Time Series Forecasts. URL https://github.com/ellisp/forecastHybrid

Shawky, D.M., Ali, A.F., 2010. A practical measure for the agility of software development processes. ICCTD 2010 - 2010 2nd Int. Conf. Comput. Technol. Dev. Proc. 230–234. https://doi.org/10.1109/ICCTD.2010.5645881

Shen, B., Ju, D., 2007. On the measurement of agility in software process, in: International Conference on Software Process. Springer Verlag, pp. 25–36. https://doi.org/10.1007/978-3-540-72426-1_3

Sherdil, K., Madhavji, N.H., 1996. Human-oriented improvement in the software process. Springer, Berlin, Heidelberg, pp. 144–166. https://doi.org/10.1007/BFb0017741

Solingen, R. van, Basili, V., Caldiera, G., Rombach, H.D., 2002. Goal Question Metric (GQM) Approach. Encycl. Softw. Eng. https://doi.org/10.1002/0471028959.SOF142

Staron, M., Hansson, J., Feldt, R., Henriksson, A., Meding, W., Nilsson, S., Hoglund, C., 2013. Measuring and Visualizing Code Stability -- A Case Study at Three Companies, in: 2013 Joint Conference of the 23rd International Workshop on Software Measurement and the 8th International Conference on Software Process and Product Measurement. IEEE, pp. 191–200. https://doi.org/10.1109/IWSM-Mensura.2013.35

Staron, M., Meding, W., 2011. Monitoring bottlenecks in agile and lean software development projects - A method and its industrial use, in: Proceedings of the 12th International Conference on Product Focused Software Process Improvement, PROFES ’11. Springer, Berlin, Heidelberg, pp. 3–16. https://doi.org/10.1007/978-3-642-21843-9_3

Staron, M., Meding, W., 2009. Using models to develop measurement systems: A method and its industrial use, in: International Workshop on Software Measurement. Springer, Berlin, Heidelberg, pp. 212–226.

https://doi.org/10.1007/978-3-642-05415-0_16

Staron, M., Meding, W., Hansson, J., Höglund, C., Niesel, K., Bergmann, V., 2014. Dashboards for Continuous Monitoring of Quality for Software Product under Development, in: Relating System Quality and Software Architecture. Elsevier, pp. 209–229. https://doi.org/10.1016/B978-0-12-417009-4.00008-9

Staron, M., Meding, W., Karlsson, G., Nilsson, C., 2011. Developing measurement systems: an industrial case study. J. Softw. Maint. Evol. Res. Pract. 23, 89–107. https://doi.org/10.1002/SMR.470

Staron, M., Meding, W., Niesel, K., Abran, A., 2017. A key performance indicator quality model and its industrial evaluation. Proc. - 26th Int. Work. Softw. Meas. IWSM 2016 11th Int. Conf. Softw. Process Prod. Meas. Mensura 2016 170–179. https://doi.org/10.1109/IWSM-MENSURA.2016.033

Staron, M., Meding, W., Palm, K., 2012. Release Readiness Indicator for Mature Agile and Lean Software Development Projects, in: Proceedings of the 13th International Conference on Agile Software Development, XP '12. Springer, Berlin, Heidelberg, pp. 93–107. https://doi.org/10.1007/978-3-642-30350-0_7

Staron, M., Meding, W., Tichy, M., Bjurhede, J., Giese, H., Söder, O., 2018. Industrial experiences from evolving measurement systems into self-healing systems for improved availability. Softw. Pract. Exp. 48, 719–739. https://doi.org/10.1002/SPE.2522

Sunetnanta, T.T., Choetkiertikul, M., 2012. Quantitative CMMI assessment for software process quality and risk monitoring in software process Improvement. Int. J. Digit. Content Technol. its Appl. 6, 95–102. https://doi.org/10.4156/jdcta.vol6.issue21.11

Susman, G.I., Evered, R.D., 1978. An Assessment of the Scientific Merits of Action Research. Adm. Sci. Q. 23, 582. https://doi.org/10.2307/2392581

Svensson, R.B., Feldt, R., Torkar, R., 2019. The Unfulfilled Potential of Data-Driven Decision Making in Agile Software Development, in: Proceedings of the 20th International Conference on Agile Software Development, XP '19. Springer, Cham, pp. 69–85. https://doi.org/10.1007/978-3-030-19034-7_5

Syed-Mohamad, S.M., Md. Akhir, N.S., 2019. SoReady: An Extension of the Test and

Defect Coverage-Based Analytics Model for Pull-Based Software Development, in: 2019 26th Asia-Pacific Software Engineering Conference (APSEC). IEEE, pp. 9–14. https://doi.org/10.1109/APSEC48747.2019.00011

Tadeusiewicz, R., 1995. Neural networks: A comprehensive foundation. Control Eng. Pract. 3, 746–747. https://doi.org/10.1016/0967-0661(95)90080-2

Tang, J.F., 2008. An adaptive model of health diagnosis for agile software development. Proc. 7th Int. Conf. Mach. Learn. Cybern. ICMLC 2, 655–659. https://doi.org/10.1109/ICMLC.2008.4620486

Tarhan, A., Demirors, O., 2012. Apply quantitative management now. IEEE Softw. 29, 77–85. https://doi.org/10.1109/MS.2011.91

Tarhan, A., Yilmaz, S.G., 2014. Systematic analyses and comparison of development performance and product quality of Incremental Process and Agile Process. Inf. Softw. Technol. 56, 477–494. https://doi.org/10.1016/J.INFSOF.2013.12.002

Taylor, S.J., Letham, B., 2018. Forecasting at Scale. Am. Stat. 72, 37–45. https://doi.org/10.1080/00031305.2017.1380080

Tüysüz, F., Kahraman, C., 2006. Project risk evaluation using a fuzzy analytic hierarchy process: An application to information technology projects. Int. J. Intell. Syst. 21, 559–584. https://doi.org/10.1002/INT.20148

Vasilescu, B., Yu, Y., Wang, H., Devanbu, P., Filkov, V., 2015. Quality and productivity outcomes relating to continuous integration in GitHub, in: Proceedings of the 10th Joint Meeting on Foundations of Software Engineering, ESEC/FSE ’15. ACM Press, New York, New York, USA, pp. 805–816. https://doi.org/10.1145/2786805.2786850

Vassallo, C., Bacchelli, A., Palomba, F., Gall, H.C., 2018. Continuous code quality: Are we (really) doing that?, in: ASE 2018 - Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering. Association for Computing Machinery, Inc, pp. 790–795. https://doi.org/10.1145/3238147.3240729

Wagner, K.W., Dürr, W., 2006. A five-step method for value-based planning and monitoring of systems engineering projects. Proc. - 32nd Euromicro Conf. Softw.

Eng. Adv. Appl. SEAA 282–290. https://doi.org/10.1109/EUROMICRO.2006.7

Wagner, S., Goeb, A., Heinemann, L., Kläs, M., Lampasona, C., Lochmann, K., Mayr, A., Plösch, R., Seidl, A., Streit, J., Trendowicz, A., 2015. Operationalised product quality models and assessment: The Quamoco approach. Inf. Softw. Technol. 62, 101–123. https://doi.org/10.1016/j.infsof.2015.02.009

Wahyudin, D., Tjoa, A.M., 2007. Event-based monitoring of open source software projects. Proc. - Second Int. Conf. Availability, Reliab. Secur. ARES 2007 1108–1115. https://doi.org/10.1109/ARES.2007.84

Wieringa, R.J., 2014. Design Science Methodology for Information Systems and Software Engineering, Design Science Methodology: For Information Systems and Software Engineering. Springer Berlin Heidelberg, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-43839-8

Wohlin, C., Runeson, P., 2021. Guiding the selection of research methodology in industry–academia collaboration in software engineering. Inf. Softw. Technol. 140, 106678. https://doi.org/10.1016/J.INFSOF.2021.106678

Wu, C.S., Chang, W.C., Sethi, I.K., 2009. A metric-based multi-agent system for software project management. Proc. 2009 8th IEEE/ACIS Int. Conf. Comput. Inf. Sci. ICIS 2009 3–8. https://doi.org/10.1109/ICIS.2009.105

Wu, H.L., Zhong, Y., Chen, Y., 2010. A software reliability prediction model based on benchmark measurement. ICIME 2010 - 2010 2nd IEEE Int. Conf. Inf. Manag. Eng. 3, 131–134. https://doi.org/10.1109/ICIME.2010.5478245

Yang, Y., Wang, Q., Li, M., 2009. Process trustworthiness as a capability indicator for measuring and improving software trustworthiness, in: International Conference on Software Process. Springer, Berlin, Heidelberg, pp. 389–401. https://doi.org/10.1007/978-3-642-01680-6_35

Yin, R.K., 2009. Case study research: Design and methods, 4th ed. Thousand Oaks, CA: SAGE Publications.

Zhang, D., Han, S., Dang, Y., Lou, J.G., Zhang, H., Xie, T., 2013. Software analytics in practice. IEEE Softw. 30, 30–37. https://doi.org/10.1109/MS.2013.94

Zhang, H., Shu, F., Yang, Y., Wang, X., Wang, Q., 2010. A fuzzy-based method for evaluating the trustworthiness of software processes, in: International Conference on Software Process. Springer, Berlin, Heidelberg, pp. 297–308. https://doi.org/10.1007/978-3-642-14347-2_26

Zhang, L., Li, L., Gao, H., 2008. 2-D software quality model and case study in software flexibility research, in: Proceedings of the International Conference on Computational Intelligence for Modelling Control and Automation, CIMCA ’08. IEEE, pp. 1147–1152. https://doi.org/10.1109/CIMCA.2008.70

Zhang, Z., Rao, G., Cao, J., Zhang, L., 2014. Software process risk measurement model based on Bayesian network. Proc. IEEE Int. Conf. Softw. Eng. Serv. Sci. ICSESS 41–44. https://doi.org/10.1109/ICSESS.2014.6933510

Annex: Interview Instrument

1. Semi-structured interview conducted at Q-Rapids industrial partners' premises

Less than ⌛ 5 Minutes
Elapsed 🕒 0

1. WARM-UP QUESTIONS:

Q1.1: **Explain your role in the company?** [We expect to interview decision-makers regarding the strategic vision of the company]

- ✓ *What are the decision-making tasks associated to your role in the company?*
- ✓ *How long have you been working in the company?*
- ✓ *How your decisions affect the software products of the company?*
- ✓ *What is your professional background? (e.g., management, informatics….)*
- ✓ *Do you have any experience in software development? If yes how long and in what roles? [Experience in traditional, agile and rapid software development before working in this company]*
  - ○ *How many years of experience in agile and rapid software development?*
  - ○ *In what roles?*

Less than ⌛ 15 Minutes
Elapsed 🕒 5

2. STRATEGIC GOALS OF THE COMPANY:

Q2.1 Which are the main STRATEGIC GOALS[40] of the company?
Q2.2 Which roles define and manage the STRATEGIC GOALS of the company?

- ✓ *How do you interact with these roles?*

Q2.3 How the success of the company STRATEGIC GOALS is measured? (STRATEGIC INDICATORS)

- ✓ *Do you use indicators? Which ones?*
- ✓ *How are these indicators measured?*
- ✓ *How are these indicators related to the Product (MODELIO) goals? Which values indicate the success or failure of the STRATEGIC GOALS of the company?*

[40] Strategic goals lead decisions in the different level of decision-making processes

- ✓ *Which roles define and manage these indicators?*

3. STRATEGIC GOALS OF THE PRODUCT (Modelio):

Q3.1 Which are the PRODUCT STRATEGIC GOALS?

Q3.2 Which roles define and manage the PRODUCT STRATEGIC GOALS?

- ✓ *How do you interact with these roles?*

  Q3.3. How is the success/failure of the software product measured (Modelio product line or a sub-product in the product line)? (PRODUCT INDICATORS)
- ✓ *Do you use some KPIs? Which ones?*
- ✓ *Are these KPI related to the Quality Requirements (QR)?*
- ✓ *How are these KPI (included QR) measured?*
- ✓ *How are they related to the PRODUCT STRATEGIC GOALS? Which values indicates the success or failure of the PRODUCT STRATEGIC GOALS?*

Less than
⌛ **15**
Minutes
Elapsed
🕒 **20**

4. RELATION BETWEEN STRATEGIC GOALS OF THE COMPANY AND GOALS OF THE PRODUCT (Modelio):

Q4.1 Which STRATEGIC GOALS of the company are related to the PRODUCT STRATEGIC GOALS?

- ✓ *In your opinion, which are the most important STRATEGIC GOALS of the company for the product? Why?*
- ✓ *How is the relationship among Strategic goals of the company and the goals of the product stated?*

Q4.2 How the STRATEGIC INDICATORS of the company are related to the PRODUCT STRATEGIC GOALS?

- ✓ *How is this relationship stated?*

Q4.3 How the product success (PRODUCT INDICATORS) is related to the STRATEGIC GOALS of the company?

- ✓ *How is this relationship stated?*

Q4.4 How the product success (PRODUCT INDICATORS) is related to the STRATEGIC INDICATORS of the company?

- ✓ *How is this relationship stated?*

Less than
⌛ **10**
Minutes
Elapsed
🕒 **35**

5. TOOLS OR PROCESSES CURRENTLY USED IN THE COMPANY

Q5.1 What tools/processes are used for dealing with the company and the Product STRATEGIC GOALS?

Q5.2 What tools/processes are used for dealing with the STRATEGIC INDICATORS of the company and the PRODUCT INDICATORS?

Q5.3 Currently, do you use any dashboard or any other tools for planning, monitoring, and controlling the projects?

If yes

- ✓ *What kind of information is provided/used by the current dashboard/tools? Where is the information coming from? Do you miss some information?*
- ✓ *What are the strengths and drawbacks of the current dashboard/tools?*
- ✓ *Who uses these tools?*

Less than
⌛ **10**
Minutes
Elapsed
🕒 **45**

6. Q-RAPIDS EXPECTATIONS

Q6.1 Which information do you think that would be valuable (and when) to improve your decision-making process regarding quality requirements (for your software development process and for your company strategic processes?

- ✓ *How to relate such valuable information with your company strategic decisions?*
- ✓ *How to relate such valuable information with your software development processes?*

Q6.2 Which roles of the company are the target user of the Q-Rapids Dashboard tool?

Q6.3 What are the main expectations of the Q-Rapids Dashboard tool?

- ✓ *Which are the most important aspects (functionalities) that you think that would be required from the expected Q-Rapids Dashboard? Why? (e.g., What-if analysis, other mitigation-strategies used in the company?)*
- ✓ *Which are the roles that would need these aspects?*
- ✓ *What tools used by managers, developers, etc., are envisaged to be connected to the Q-Rapids dashboard (e.g., Sonar)?*
- ✓ *Is there any preference about the user interface design of the Q-Rapids Dashboard?*
- ✓ *What are the main challenges that you think that are related to Q-Rapids Dashboard and tools?*
- ✓ *What type of reports would you like Q-Rapids to provide?*

Less than
⏳ 5
Minutes

Elapsed
🕒 55

7. ADDITIONAL COMMENTS:

Q7.1: Are there **any related issues that we missed** and that you would like to reflect on?

Appendix 1: Details of the SESSI Tool Support

The following sections provide detail on the software artifacts developed to provide tool support to the different steps of the SESSI method.

1. SESSI Phase 2: DAG Specification

As a support for this step, we implemented the equal-width and the equal-frequency algorithms as software artifacts in order to assist the specification of the binning intervals. These artifacts are implemented in Java and are freely available and open source in GitHub[41]. For the case of the equal-frequency binning, it is dependent on the provided historical data (i.e., training set) to produce the equally sized bins. The artifact is prepared to take the historical data from a specified Elasticsearch database index. However, as the artifact is open source, it can be adapted to gather the historical data from a different database or local file.

Both algorithms are briefly described and illustrated with examples as follows:

[41] https://github.com/martimanzano/SSI-assessment/blob/TH/src/main/java/Util_Assessment_SI/BayesUtils.java#L213

- **Equal-Width binning:** Given a specified desired number of bins *k* equal to the number of node's states, this algorithm divides the numeric interval [min, …, max] into *k* bins of equal *w* size:

$$w = {}^{(max - min)}\!/_{k}$$

For example, the equal-width algorithm applied to a node with 5 ordinal states {*Very Low, Low, Medium, High, Very High*} and whose values are computed from data yielding in the numerical interval [0, 1] would return 5 bins, each one with width *w* = 0.2, i.e. [0-0.2), [0.2-0.4), [0.4-0.6), [0.6-0.8), [0.8-1]. This algorithm is independent of the values existing in the historical data of the variable/s to discretize and should be used when the interval size should be uniform.

- **Equal-Frequency binning:** Given a specified desired number of bins *k* equal to the number of node's states, this algorithm divides the numeric interval [*min*, …, *max*] into *k* bins such as each one would contain approximately an equal number of discretized data points. Hence this algorithm depends on the actual collected numerical values in the historical data and can be used when the intervals should have roughly the same size of binned values (i.e., when the distribution of the binned values should be approximately uniform). For example, the equal-frequency algorithm applied to a node with 5 ordinal states {*Very Low, Low, Medium, High, Very High*}, computed from data yielding in the numerical interval [0, 1] and with 10 collected data points as input {0, 0.1, 0.3, 0.4, 0.6, 0.6, 0.85, 0.91, 0.98, 0.99}, would return 5 bins, i.e. [0, 0.2), [0.2, 0.5), [0.5, 0.725), [0.725, 0.945), [0.945, 1]. Applying the binning function to each data point, each bin would contain the same number of data points (i.e., 2):

-Bin 1: 0, 0.1
-Bin 4: 0.85, 0.91
-Bin 2: 0.3, 0.4
-Bin 5: 0.98, 0.99
-Bin 3: 0.6, 0.6

2. SESSI Phase 2: CPTs Specification

We implemented a set of software artifacts to support the probability elicitation and therefore the construction of the nodes' Conditional Probability Tables (CPTs).

For the metric nodes, we developed a software artifact named *getFrequencyQuantification* [42] to perform the frequency quantification automatically. This artifact takes as input the specified states and binning functions for a specific node. The software artifact queries an Elasticsearch index containing the historical data for the node under quantification. Afterwards, it discretizes every numerical value of the given dataset according to the specified binning functions, and finally it computes the relative quantification of each state.

For child nodes, we have developed an implementation of the Weighted Sum Algorithm (WSA)[43]. Additionally, to ease the specification of the WSA's required input, we developed a data-driven software artifact that computes the compatible configurations required by the WSA automatically from the historical data. Such software artifact is named *getCompatibleConfigurations* and is open source and available in GitHub[44]. As the other artifacts, it is prepared to query the required data from a given Elasticsearch index.

When the parent nodes of the child node under quantification are metric nodes, i.e., the child node *Code Quality* in the example *Product Quality* SSI estimation model, the software artifact computes the compatible parental configurations from the provided historical data. It uses the states and binning functions specified in the previous step per parent node, grouping and ranking the observed combination of parent nodes' states according to the frequency of such coexistence. However, there is a special case requiring consideration. That is, when the parent nodes of the child node under quantification are not on the bottom level of the Bayesian Network (BN) (i.e., not metrics), and hence cannot be directly computed from the numerical values of the historical data, as they yield only in the BN, like the *Product Quality* node in the example SSI estimation model shown in Chapter 6. For these cases, we developed a variant of the artifact *getCompatibleConfigurations* named

[42] https://github.com/martimanzano/SSI-assessment/blob/TH/src/main/java/Util_Assessment_SI/BayesUtils.java#L190

[43] https://git.io/Jvspi

[44] https://github.com/martimanzano/SSI-assessment/blob/TH/src/main/java/Util_Assessment_SI/BayesUtils.java#L18

getChildCompatibleConfigurations[45]. The mentioned artifact can compute the compatible parental configurations propagating the probabilities to the parent nodes of the node under quantification. Therefore, it is necessary to construct a partial BN containing the parent nodes and their CPTs before using such *getChildCompatibleConfigurations* artifact.

With the use of the two described software artifacts, the effort required for specifying the inputs required by the WSA gets significantly reduced, and hence the CPTs filling process for child nodes can be achieved semi-automatically.

3. SESSI Phase 2: Estimation Model Validation

The validation scenarios to use in both validations described in Chapter 6 can be entered into the SSI estimation model with the BN software (i.e., Netica or unBBayes) used during the Estimation Model Generation step. To specify the scenarios to use in the Outcome Adequacy, the software artifacts *getCompatibleConfigurations* and *getChildCompatibleConfigurations* may be used to perform the specification automatically, as these tools extract combinations of parent states for the node under construction or validation from the historical data. These artifacts have been already described in section 2 (SESSI Phase 2: CPTs Specification).

4. SESSI Phase 3: SSI Monitoring

To support and ease the deployment of the SSI estimation model and its connection to the data collectors, we developed the *SSI Estimation* component mentioned in Chapters 7 and 8. This component was developed as an open-source Java library[46]. Such software library can be deployed or embedded in existing architectures. This library acts as a wrapper of the unBBayes API, which performs BN inference on BNs created either with the unBBayes or Netica® GUIs. It uses the unBBayes implementation of the junction tree algorithm to perform the BN inference. Additionally, the library may be

[45] https://github.com/martimanzano/SSI-assessment/blob/TH/src/main/java/Util_Assessment_SI/BayesUtils.java#L45

[46] SSI-Assessment (https://github.com/martimanzano/SSI-assessment)

used as a REST web service, allowing its use independently from the programming language used by the software orchestrator or the software to embed the library.

Appendix 2: Details of the Case Study from Chapter 8

1. CPT Elicitation and validation tables

Table A 1 Elicited states, binning intervals, and quantified probabilities for each metric node

Development Task Completion		
State	**Interval**	**Quantified freq.**
VeryLow	[0, 0.45)	30%
Low	[0.45, 0.7)	20%
Medium	[0.7, 0.90)	20%
High	[0.90, 0.95)	10%
VeryHigh	[0.95, 1]	20%
Specification Task Completion		
State	**Interval**	**Quantified freq.**
VeryLow	[0, 0.2)	15%
Low	[0.2, 0.70)	30%
Medium	[0.70, 0.90)	15%
High	[0.90, 0,99)	10%
VeryHigh	[0.99, 1]	30%
Postponed Issues (Closed) Ratio		
State	**Interval**	**Quantified freq.**
Low	[0, 0.45)	50%
Medium	[0.45, 0.80)	40%
High	[0.80, 1]	10%
Build Stability		
State	**Interval**	**Quantified freq.**
VeryLow	[0, 0.4)	3%
Low	[0.4, 0.7)	4%

Medium	[0.7, 0.8)	8%
High	[0.8, 0.95)	25%
VeryHigh	[0.95, 1]	60%
Critical Issues (Closed) Ratio		
State	**Interval**	**Quantified freq.**
VeryLow	[0, 0.4)	3%
Low	[0.4, 0.70)	4%
Medium	[0.70, 0.80)	30%
High	[0.80, 0.98)	55%
VeryHigh	[0.98, 1]	8%
Passed Tests Percentage		
State	**Interval**	**Quantified freq.**
VeryLow	[0, 0.4)	15%
Low	[0.4, 0.70)	10%
Medium	[0.70, 0.80)	5%
High	[0.80, 0.98)	40%
VeryHigh	[0.98, 1]	30%

Table A 2 CPT for the factor node *Known Remaining Defects (Closed) Ratio*

Metric	**Known Remaining Defects (Closed) Ratio**		
Postponed Issues (Closed) Ratio	**Low (%)**	**Medium (%)**	**High (%)**
Low	90	5	5
Medium	5	90	5
High	5	5	90

Table A 3 Partial CPT elicited for the factor node *Product Stability*, along with the relative weights required by the WSA algorithm

Metrics			**Product Stability**				
Build Stability (W=20%)	**Critical Issues (Closed) Ratio (W=40%)**	**Passed Tests Percentage (W=40%)**	**Very Low (%)**	**Low (%)**	**Medium (%)**	**High (%)**	**Very High (%)**
VeryLow	VeryLow	VeryLow	90	10	0	0	0
VeryLow	Low	VeryLow	70	20	10	0	0

Low	Low	Low	55	40	5	0	0
Medium	High	Medium	3	25	50	20	2
High	Medium	High	3	25	50	20	2
High	VeryHigh	VeryHigh	0	3	7	20	70
VeryHigh	VeryLow	VeryHigh	8	10	62	15	5
VeryHigh	Low	Low	4	30	50	15	1
VeryHigh	Low	VeryHigh	6	7	65	17	5
VeryHigh	Medium	High	2	18	53	22	5
VeryHigh	Medium	VeryHigh	1	15	55	20	10
VeryHigh	High	High	2	3	20	45	30
VeryHigh	High	VeryHigh	1	2	22	40	35
VeryHigh	VeryHigh	High	0	5	15	60	20
VeryHigh	VeryHigh	VeryHigh	0	1	4	15	80

Table A 4 Partial CPT elicited for the factor node *Activity Completion*, along with the relative weights required by the WSA algorithm

Metrics		**Activities Completion**				
Development Task Completion (W=70%)	**Specification Task Completion (W=30%)**	**Very Low (%)**	**Low (%)**	**Medium (%)**	**High (%)**	**Very High (%)**
VeryLow	VeryLow	90	10	0	0	0
VeryLow	Low	80	15	5	0	0
VeryLow	Medium	45	30	20	5	0
VeryLow	VeryHigh	25	35	20	17	3
Low	VeryHigh	25	35	25	11	4
Medium	Medium	5	20	55	20	0
Medium	VeryHigh	6	9	40	35	10
High	High	3	7	15	60	15
High	VeryHigh	3	7	15	55	20
VeryHigh	VeryHigh	1	2	7	20	70

Table A 5 Partial CPT elicited for the SSI node *Product Readiness*, along with the relative weights required by the WSA algorithm

Factors			**Product Readiness**			
Activities Completion (W=60%)	**Known Remaining Defects (Closed) Ratio (W=10%)**	**Product Stability (W=30%)**	**Not Ready (%)**	**Neutral (%)**	**Almost Ready (%)**	**Ready (%)**
VeryLow	Low	VeryLow	98	2	0	0
VeryLow	Low	Low	90	10	0	0
VeryLow	Low	High	65	30	4	1
VeryLow	Medium	Medium	55	40	4	1
VeryLow	Medium	High	50	45	4	1
Low	Low	Low	75	25	0	0
Low	Medium	Medium	48	45	6	1
Low	Medium	High	45	50	3	2
Medium	Low	Medium	40	50	8	2
Medium	Medium	Medium	15	70	10	5
Medium	Medium	High	8	60	30	2
High	Medium	Medium	2	40	55	3
VeryHigh	Medium	Medium	1	19	60	20
VeryHigh	High	High	1	4	20	75
VeryHigh	High	VeryHigh	1	4	5	90

Table A 6 Model Walkthrough validation data for the *Activities Completion* node

Metrics		**Activities Completion**	
Development Task Completion	**Specification Task Completion**	**Participants' precepted most probable state**	**BN Output**
VeryLow	High	VeryLow	🟥 Low
Low	VeryLow	Low	🟩 VeryLow
Low	Low	Medium	🟩 VeryLow
Low	High	Medium	🟥 Low
Medium	Low	Medium	🟥 Low
Medium	High	VeryLow	🟩 Medium
High	Medium	VeryLow	🟥 Medium

Medium	Medium	VeryLow	Medium
VeryHigh	High	High	High
VeryHigh	Medium	High	High
High	Low	Low	Medium
VeryHigh	VeryHigh	High	VeryHigh

Table A 7 Model Walkthrough validation data for the *Known Remaining Defects* node

Metric	**Known Remaining Defects (Closed) Ratio**	
Postponed Issues (Closed) Ratio	**Participants' precepted most probable state**	**BN Output**
High	High	High

Table A 8 Model Walkthrough validation data for the *Product Stability* node

Metrics			**Product Stability**	
Build Stability	**Critical Issues (Closed) Ratio**	**Passed Tests Percentage**	**Participants' precepted most probable state**	**BN Output**
VeryLow	VeryLow	Low	VeryLow	VeryLow
Low	Medium	Low	Low	Medium
Medium	VeryHigh	Medium	Medium	Medium
VeryHigh	Low	High	Medium	Medium
VeryHigh	VeryHigh	Medium	Medium	Medium
VeryHigh	VeryLow	VeryLow	VeryLow	VeryLow
VeryHigh	Low	VeryLow	VeryLow	VeryLow
High	Medium	VeryLow	VeryLow	VeryLow
Medium	VeryHigh	High	High	High
Low	VeryHigh	VeryHigh	High	VeryHigh
Low	High	Low	Low	Low
High	High	High	High	High
High	VeryHigh	High	High	VeryHigh
VeryHigh	VeryHigh	Low	Low	VeryHigh

Table A 9 Model Walkthrough validation data for the *Product Readiness* node

Factors			**Product Readiness**	
Activities Completion	**Known Remaining Defects (Closed Ratio)**	**Product Stability**	**Participants' precepted most probable state**	**BN Output**
VeryLow	Low	Medium	NotReady	NotReady
VeryLow	Medium	VeryLow	NotReady	NotReady
Medium	Low	High	Neutral	Neutral
Medium	Medium	VeryHigh	Neutral	Neutral
High	High	Medium	AlmostReady	AlmostReady
High	High	Low	Neutral	AlmostReady
VeryHigh	High	Medium	Neutral	Ready
VeryHigh	High	VeryLow	AlmostReady	Ready
VeryHigh	Low	High	AlmostReady	Ready
Medium	High	VeryHigh	Neutral	Neutral
Medium	Low	VeryHigh	Neutral	Neutral
Medium	Low	VeryLow	NotReady	NotReady
Low	High	Low	NotReady	NotReady
Low	Low	Medium	NotReady	NotReady

Table A 10 Outcome Adequacy validation data for the *Activities Completion* node

Metrics		**Activities Completion**	
Development Task Completion	**Specification Task Completion**	**Assessed state**	**BN Output**
VeryHigh	VeryHigh	VeryHigh	VeryHigh

Table A 11 Outcome Adequacy validation data for the *Known Remaining Defects* node

Metric	Known Remaining Defects	
Postponed Issues (Closed) Ratio	Assessed state	BN Output
VeryHigh	VeryHigh	VeryHigh

Table A 12 Outcome Adequacy validation data for the *Product Stability* node

Metrics			Product Stability	
Build Stability	Critical Issues (Closed) Ratio	Passed Tests Percentage	Assessed state	BN Output
VeryHigh	Medium	VeryHigh	Medium	Medium
VeryHigh	Medium	High	Medium	Medium
VeryHigh	Medium	VeryHigh	Medium	Medium
High	Medium	VeryHigh	Medium	Medium
VeryHigh	Medium	VeryHigh	Medium	Medium
VeryHigh	Medium	VeryHigh	Medium	Medium
VeryHigh	Medium	VeryHigh	VeryHigh	Medium
VeryHigh	High	VeryHigh	High	High
High	High	VeryHigh	High	Medium
VeryHigh	High	VeryHigh	VeryHigh	High

Table A 13 Outcome Adequacy validation data for the *Product Readiness* node

Factors			Product Readiness	
Activities Completion	Known Remaining Defects (Closed Ratio)	Product Stability	Assessed state	BN Output
VeryHigh	Medium	Medium	AlmostReady	AlmostReady
VeryHigh	Medium	High	Ready	Ready
VeryHigh	Medium	Medium	Ready	AlmostReady

2. Questionnaire conducted in the Case Study

1. Score the method providing a response scale from -2 to 2 for each of the following adjectives

:(	-2	-1	0	+1	+2	:)
Useless						**Useful**
Incomplete						**Complete**
Unreliable						**Reliable**
Incomprehensible						**Comprehensible**
Unclear						**Clear**
Irrelevant						**Relevant**
Ambiguous						**Self-explanatory**
Abstract						**Detailed**
Complex						**Simple**
Tedious						**Interesting**
Slow						**Rapid**
Inefficient						**Efficient**
Difficult						**Easy**
Long						**Short**
Unrepeatable						**Repeatable**

2. Write up to the three aspects you like the most of the method and three aspects you would like to improve/change/eliminate in the method and explain the reason.

Aspects you like the most of the method	**Explanation**

Aspects you would like to improve/change/eliminate in the method	**Explanation**

3. Do you think you would be able to repeat the steps of the method by yourself? (Having the help of a user guide)