Quantifying Inner Source Software Development for Business Process Usage

Quantifizierung von Inner Source Softwareentwicklung für die Nutzung in Geschäftsprozessen

Der technischen Fakultät
der Friedrich-Alexander-Universität
Erlangen-Nürnberg

zur

Erlangung des Doktorgrades
Doktor-Ingenieur (Dr.-Ing.)

vorgelegt von

Stefan Buchner
aus Neumarkt i.d.Opf.

Als Dissertation genehmigt
von der Technischen Fakultät
der Friedrich-Alexander-Universität Erlangen-Nürnberg

Tag der mündlichen Prüfung: 10.05.2024
Gutachter: Prof. Dr. Dirk Riehle
Prof. Dr. Florian Matthes

Abstract

Inner source refers to the application of open source development principles within corporate environments. Essentially, it involves embracing open collaboration and contribution without necessarily developing open source software. One fundamental concept borrowed from open source development is the practice of opening software projects to external contributions. In the realm of inner source, these contributions usually originate from developers engaged in other projects or working in different organizational units. This collaborative approach yields mutual benefits through joint software development.

However, these inner source contributions often cross organizational and legal boundaries, significantly impacting a wide range of business processes. These effects extend beyond software development and affect strategic and operational functions across various departments, including management, accounting, and even taxation. Currently, these processes are not harmonized with the inner source way of intellectual property flow within the company. This misalignment can lead to mismanagement and even profit shifting. The primary objective of this dissertation is to address these challenges by making three consecutive contributions:

First, we conducted a systematic literature review, combined with thematic analysis, to assess the current state of inner source measurement and its impact on various business processes. This qualitative data analysis categorizes a range of approaches and evaluates their suitability for application within the inner source domain.

Second, building on the insights derived from our systematic literature review, we developed an inner source research model. This model serves as a foundation, providing researchers with a unified framework for advancing toward more precise and comprehensive inner source measurement. It, in turn, enables the development of future tools and techniques and facilitates the creation of metrics tailored to inner source management, aligning existing business processes with the inner source paradigm.

Third, we have implemented a first algorithm for the financial assessment of inner source contributions. This algorithm estimates the time invested in code contributions by individual contributors or departments. These estimations, in turn, enable cost calculations that can be applied to business processes affected by inner source, such as tax contribution assessments.

Lastly, we consolidate individual contributions, clarify thematic connections, integrate outside research, and explore future prospects. In summary, this dissertation lays the theoretical groundwork for the financial evaluation of inner source contributions. It offers guidance to researchers on conducting measurement-related inner source research and provides practitioners with insights into the development of inner source metrics and tools. Furthermore,

this dissertation introduces an initial algorithm, practically evaluated for its usability in conducting cost calculations within the inner source domain.

Zusammenfassung

Inner Source ist die Anwendung von Open-Source Prinzipien innerhalb von Unternehmen. Im Wesentlichen werden die Prinzipien der offenen Zusammenarbeit genutzt, ohne tatsächlich Open-Source-Software zu entwickeln. Ein grundlegendes Konzept aus der Open-Source-Entwicklung ist die Praxis, Softwareprojekte für externe Beiträge zu öffnen. Im Inner Source Bereich stammen diese Beiträge von Entwicklern, die an anderen Projekten beteiligt sind oder in verschiedenen Organisationsbereichen sitzen. Dieser kollaborative Ansatz führt durch die gemeinsame Entwicklung von Software zu gemeinsamen Vorteilen.

Allerdings überschreiten diese Inner Source Beiträge oft organisatorische und rechtliche Grenzen. Dieses Prinzip beeinflusst maßgeblich einen weiten Bereich von Geschäftsprozessen, der über die Softwareentwicklung selbst hinausgeht und sich auf strategische und operative Funktionen erstreckt, einschließlich Management, Buchhaltung und dem Steuerwesen. Derzeit sind diese Prozesse nicht im Einklang mit der Inner Source Art des IP-Flusses innerhalb des Unternehmens. Diese Abweichung kann zu Fehlmanagement und sogar Gewinnverschiebungen führen. Das Ziel dieser Dissertation ist es, diesen Herausforderungen durch drei aufeinanderfolgende Beiträge zu begegnen:

Erstens führten wir eine systematische Literaturanalyse in Verbindung mit einer thematischen Analyse durch, um den aktuellen Stand der Inner Source Messung und deren Auswirkungen auf verschiedene Geschäftsprozesse zu ermitteln. Diese qualitative Datenanalyse kategorisiert eine Reihe von Ansätzen und bewertet deren Eignung für die Anwendung im Bereich Inner Source.

Zweitens bauen wir auf den Erkenntnissen unserer systematischen Literaturanalyse ein Forschungsmodell für Inner Source Messung auf. Dieses Modell dient als Grundlage und bietet den Forschern einen einheitlichen Rahmen für die Entwicklung einer genaueren und umfassenderen Messung von Inner Source Entwicklung. Dadurch wird eine Vielzahl zukünftiger Werkzeuge und Techniken ermöglicht. Darüber hinaus wird die Erstellung von Kennzahlen, die auf das Inner Source Management zugeschnitten sind, und die Anpassung bestehender Geschäftsprozesse an das Inner Source Paradigma erleichtert.

Drittens haben wir einen ersten Algorithmus für die finanzielle Bewertung von Beiträgen im Inner Source implementiert. Dieser Algorithmus schätzt die Zeit, die in Codebeiträge von einzelnen Entwicklern oder Abteilungen investiert wird. Diese Schätzungen ermöglichen wiederum Kostenberechnungen, die auf von Inner Source betroffene Prozesse angewendet werden können, wie z. B. Berechnung von Steuerbeiträgen.

Schließlich konsolidieren wir die individuellen Beiträge und klären thematische Verbindungen sowie die Integration externer Forschung und zukünftiger Beiträge. Zusammenfassend

legt diese Dissertation die theoretische Grundlage für die finanzielle Bewertung von Inner Source Beiträgen fest. Sie bietet Forschern eine Grundlage zur Durchführung von Inner Source Forschung im Bereich der Messung und gibt Praktikern Einblicke in die Entwicklung von Inner Source Metriken und -Tools. Darüber hinaus führt diese Dissertation einen ersten Algorithmus ein, der praktisch auf seine Verwendbarkeit bei der Durchführung von Kostenberechnungen im Inner Source Bereich evaluiert wurde.

Contents

List of Figures

List of Tables

List of Acronyms

IP	Intellectual Property
KPI	Key Performance Indicators
LoC	Lines of Code
OECD	Organisation for Economic Co-operation and Development
SLR	Systematic Literature Review

Acknowledgments

First and foremost, I would like to thank my supervisor Prof. Dirk Riehle for his continuous guidance and support throughout this journey. His initiative in setting up our project group and facilitating our work on practical problems has been immensely valuable. I also would like to thank Prof. Florian Matthes as the second reviewer of this dissertation.

I am very grateful to be able to work with Julian H., Thomas, Georg, and Philip over the years. Our productive discussions during both office meetings and online interactions have significantly improved my work and have also been a lot of fun. I am thankful to Elcin, Julian L., Julia, Andreas, Ivone, Anna, and Fariba for their support and feedback on various research and bureaucratic matters.

I would also like to thank the DFG and BMBF's (Federal Ministry of Education and Research) SoftwareCampus project for partially funding my research.

A big thank you to Lukas and Michael for their consistent support and the wonderful times we've shared over the past decade. Your friendship has helped me get to where I am now. A special thanks goes to my parents, Rosi and Jürgen, and my brother Tobias, for their love, support, and guidance at every step along the way.

Related Publications

In this dissertation the third person plural (*"we"*) is used. Main reason is that this dissertation is cumulative and all the contributing papers were published in co-authorship with other researchers. To honor the contributions of the co-authors to the individual work and improve the thesis readability I will consistently use the term *"we"* within the thesis.

This cumulative dissertation consists of three main papers that I wrote as leading author with my supervisor Dirk Riehle as co-author. In all three articles I took the role as main author and therefore contributed the major part to the overall results of the paper. The following articles are part of this thesis:

- Buchner and Riehle [14]: This article contains the results of a conducted systematic literature review and thematic analysis in the business process domain and computational tools and techniques. It also includes a classification of the suitability of the identified approaches for the usage within inner source development

- Buchner and Riehle [13]: This article builds an inner source research model on the insights generated during the literature review. For the design of the research model six hypotheses were generated that can guide future researcher in their inner source measurement-related work. Goal is to improve and adapt existing business processes to the inner source paradigm by creating a comprehensive measurement basis.

- Buchner and Riehle [12]: This article presents an algorithm that estimates the time spent on code contributions for the usage within cost-related business processes. The algorithm measures the time spend on commits. The result is work time share per project that is calculated for each department.

1

Introduction

Collaborating across international and organizational boundaries is increasingly common in modern software development, particularly within large companies. This not only includes project-specific collaborations but also organization-wide programs [17], such as platform development [52]. Over the last decade, the practice of software developers collaborating across legal boundaries has been recognized as *inner source* [56].

In the context of inner source, the freedom to contribute to previously unassigned projects in a self-organized manner is a key benefit [53]. As software development grows more complex, inner source is becoming increasingly common in both tech and non-tech companies. A survey conducted by the InnerSource Commons[1], an organization that facilitates communication and knowledge sharing between companies implementing inner source, revealed that 37.8% of respondents were from the technology sector. The majority of respondents represented other sectors such as healthcare, financial services, or retail.

The increasing complexity in software development and the growing application of inner source are also influencing various business processes. This is primarily due to contributions, such as software code, moving between different legal and tax boundaries [12]. For instance, within the domain of taxation, any intellectual property (IP) flow between two legally dependent entities (e.g. within a holding company) must be treated as if it were conducted between external, independent actors [50]. This principle is commonly referred to as the arm's length principle [50]. Consequently, inner source contributions that cross legal boundaries must also adhere to this principle. This creates tax obligations that pose a significant challenge. Calculating the value of the IP, commonly known as the transfer price [47], is complex. The Organisation for Economic Co-operation and Development (OECD) has established several international guidelines for calculating the transfer price [47]. However, these guidelines are not easily applicable to digital business models, there are many complications for software development [46] that also apply to inner source [12].

With inner source, the calculation of transfer prices is getting increasingly inaccurate [12]. That impacts not only external stakeholders but also internal processes, workflows, and decision models. One important example are managerial decisions, for which first inner source applicable models were already designed [31]. Furthermore, the reduced collaboration boundaries in inner source pose a challenge for middle managers who often lack of buy-in due to their fear of loosing their performance goals [52, 1].

This dissertation addresses which business processes are influenced by inner source and proposes effective strategies for their management.

Business processes that do not fit to inner source have in common that there is currently no sufficient way how to financially assess cross-boundary collaboration. As this dissertation will demonstrate, there are only few existing tools and techniques for measuring inner source. Even fewer are suitable for economic assessments within businesses and their processes. It is unclear how to consistently assess IP related to inner source for various financial and managerial assessment processes. Additionally, the specific prerequisites and data structures necessary to facilitate a comprehensive assessment of different IP flows are not well defined.

This dissertation will extensively explore the current state of tools and techniques for measuring inner source. We will also show their applicability to various business processes influenced by inner source. Moreover, this thesis aims to address the need for improved software development measurement across a wide range of business processes. The overarching goal is to enhance business processes and related decision-making affected by cross-organizational collaboration in software development.

To achieve this goal, the dissertation's outline is as follows: Section 2 provides additional context about the current state of the art of inner source and its impact on related business processes. Section 3 outlines the goals of the thesis and the overall dissertation concept in detail. Section 4 offers an overview of the different methods employed to achieve these goals. Section 5 presents the results of the individual research papers that constitute the cumulative dissertation. That includes the outcomes of a systematic literature review, an inner source research model, and an algorithm for estimating the time spent on code contributions, which can be used for cost calculation processes. Section 6 examines how the different aspects presented in the results section complement each other and discusses their integration within external and future research. Lastly, Section 7 concludes the thesis.

2

State of the art

In the inner source domain, various topics have already been extensively researched. Several research articles since 2010 defined inner source and its underlying principles [56, 17, 27, 43]. A systematic review conducted by Edison et al. [27] revealed that a majority of inner source research focuses on theory (43.24%), lessons learned (13.51%), and frameworks (13.51%). According to their review, the bulk of previous and ongoing research looks into the fundamental theoretical aspects of inner source governance, motivations, benefits, and challenges. Efforts have been directed towards fostering collaboration and breaking down internal barriers, especially regarding communication. However, the review by Edison et al. [27] indicates that only a minority of studies developed their own models (8.11%) or tools (10.81%). Notably, most of the research in the domain of inner source is of qualitative nature.

Recent research confirmed the benefits of implementing inner source. Companies, for instance, leveraged inner source to enhance their efficiency (e.g., achieving faster time-to-market, reducing costs), promote code reuse, improve employee motivation, and overcome organizational boundaries [17]. Overall, inner source facilitates improved knowledge management and sharing capabilities [27].

Nevertheless, various challenges persist within the inner source domain. For example, the optimal management of knowledge in inner source is yet to be realized [27], which shows in the inadequate documentation for many inner source projects [56]. These challenges, as identified by Edison et al. [27], are connected to cultural issues within companies, such as resistance to change [40, 45, 18] and the need to motivate individuals to expand their expertise beyond their specific domains [43]. Additionally, concerns related to the security and transparency of internal code sharing persist among developers [27].

Furthermore, there are several non-social and non-cultural challenges associated with the general management of inner source [27]. These include the necessity to provide clear implementation guidelines to individuals [26] and the lack of effective measurement and assessment methods for inner source initiatives [16, 12, 14, 13]. This thesis, in particular, aims to contribute to the resolution of the latter issue.

In the domain of inner source measurement, limited research has been conducted beyond the scope of this thesis. Notably, Edison et al. [27] highlighted existing measurements and metrics of inner source. Their analysis revealed that the majority of research focused on quantifying the number of inner source projects [22, 53] or tracking the volume of contributions [39, 59]. A more comprehensive model was introduced by Capraro et al. [16], who developed a method for measuring the flow of contributions across different organizational units within inner source. Furthermore, Hirsch and Riehle [31] developed initial models and metrics for inner source management accounting and decision-making. However, the existing literature lacks extensive metrics for evaluating inner source, particularly from the perspective of business processes, a crucial aspect for the success of inner source initiatives.

In the broader context of business processes, well-established methodologies and models exist. Notably, the OECD extensively describes several methods for calculating transfer prices, commonly employed by its member states [47, 60]. In their research, the OECD acknowledged the risk of profit shifting due to digital business models and international software development, which current models fail to address adequately [46, 41, 12]. This thesis aims to propose metrics to address these challenges.

Additionally, well-defined business processes in accounting and management exist, including principles for cost calculation [6] and fundamental accounting principles in general [58]. Within the accounting domain, certain approaches demonstrate how established models can be adapted to collaborative development, such as platform development [37, 42].

This thesis, alongside related research forming this cumulative dissertation, offers a comprehensive analysis of the current state of inner source research, particularly focusing on its measurement, evaluation, and the associated business processes. The insights emphasize that in-depth exploration of inner source measurement for addressing business challenges can significantly improve the adoption of inner source. Subsequently, the efficiency of software development within companies can be enhanced.

3
Aim of the Thesis

Before presenting the applied methods and results of this dissertation, we first outline the key issues we seek to address. This section begins by introducing the central research question guiding the thesis. Additionally, it outlines the goals and aspects that are outside the scope of this work. Additionally, we have a look at the concept of the following sections.

3.1 Goals and Research Question

Previous sections highlighted the lack of research in the domain of inner source measurement, especially concerning the assessment of its impact on business processes. Consequently, this dissertation answers the following research question:

RQ: How can inner source software development be quantified for use within business processes?

The dissertation answers the research question by looking into different problems that are solved by research papers from which the thesis is composed of. Those play an important part in answering the overall research question. Following the research question and problem motivation we derived the following goals of this thesis:

- Identify the current state of research on the influence of inner source on business processes.
- Survey existing tools and techniques for evaluating software development efforts, particularly within the context of inner source.
- Determine how these tools and techniques can enhance business processes affected by inner source.
- Construct a comprehensive research model to facilitate a unified understanding of future research requirements in the domain of inner source quantification.
- Develop an initial algorithm for assessing inner source efforts that can be utilized for cost calculations within business processes.
- Provide a perspective on how the various insights merge and expand in current and future research.

The primary aim of this dissertation is to establish foundational knowledge for future research rather than to develop a comprehensive guide on the integration of the measured aspects into various affected business processes. The focus remains on preparing the data and facilitating the creation of metrics and models. We emphasize a technical perspective rather than providing a business process management and integration guide.

Moreover, it should be noted that this dissertation does not aim to introduce new processes, such as novel transfer pricing methods. Instead, it identifies areas within the measured domains where current processes are unsuitable for inner source applications, laying the groundwork for future research to develop new processes.

In addition, the scope of this work is focused on the measurement and financial assessment of inner source, rather than an exploration of inner source governance. Given that some affected business processes, particularly those related to taxation, rely on reproducible and transparent algorithms, the primary focus within the inner source measurement domain does not involve the application of (black box) machine learning approaches. Recognizing that inner source fosters high-frequency contributions, we also seek to enable automated solutions with minimal manual intervention.

3.2 Dissertation Concept

The structure of the dissertation involves two key aspects:

First, the key findings and insights generated by this dissertation as well as related papers are presented (Section 5). These findings serve as the foundation for the cumulative dissertation. The research papers are organized in a logical sequence, with the initial articles laying the groundwork for subsequent research. The presentation begins with the fundamentals of literature and ends in the demonstration of an algorithm designed to achieve the dissertation's objectives. Consequently, a bottom-up approach is adopted, facilitating the development of further inner source measurement algorithms, metrics, and models.

Second, the interconnections between the individual results and research papers are analyzed (Section 6). This includes an examination of their integration with external research and how future research can leverage the insights presented in this work.

Table 3.1 provides an overview of the three main papers contributing to this thesis. It outlines the sections where the respective methods are described and offers a short summary of the contents, along with the corresponding sections where the content is extensively discussed. Notably, two of the papers, the systematic literature analysis [14] and the research model [13], are based on the same two methods.

The first paper (Buchner and Riehle [14]) lays the groundwork for subsequent research by conducting a survey of existing literature on inner source. This survey explores its impact on

<table>
<tr><th></th><th colspan="2">Methodology</th><th colspan="2">Results</th></tr>
<tr><th>Paper</th><th>Name</th><th>Section</th><th>Content</th><th>Section</th></tr>
<tr><td>[14]</td><td rowspan="2">- Kitchenham [36]:
Systematic Literature Review
- Braun & Clarke [10]:
Thematic Analysis</td><td rowspan="2">4.1.1
4.1.2</td><td>- Influence of inner source on business processes
- Current tools and techniques of measuring inner source</td><td>5.1</td></tr>
<tr><td>[13]</td><td>- Research model on inner source measurement</td><td>5.2</td></tr>
<tr><td>[12]</td><td>- Peffers et al. [49]:
Design Science</td><td>4.2</td><td>- An algorithm for work time based cost calculation in inner source</td><td>5.3</td></tr>
</table>

Table 3.1: Method and Results Overview

business processes, as well as the identification of tools and techniques suitable for measuring software development effort. Our approach combined the methodology of systematic literature review by Kitchenham [36] and thematic analysis by Braun and Clarke [10]. Detailed descriptions of these methods are available in Section 4.1. The results are presented in Section 5.1.

The second paper (Buchner and Riehle [13]) introduces a research model built upon insights derived from the systematic literature review and thematic analysis. This publication consolidates various perspectives, notably demonstrating the compatibility of the computational and business process perspectives, thereby fostering enhanced inner source adoption. The results of this paper are explained in Section 5.2.

Subsequently, the third paper (Buchner and Riehle [12]) introduces an algorithm applicable to various business processes. This algorithm estimates the time invested in code contributions, facilitating fairer cost allocation across different organizational units. The development of this algorithm followed the design science process outlined by Peffers et al. [49] as an artifact-driven approach for designing and evaluating software solutions. Section 4.2 provides an in-depth presentation of this method, with the results discussed in Section 5.3.

4

Methodology

For this thesis, we employed various methodologies that align with the specific goals and research questions presented in the respective articles. Our approach included a SLR in connection with thematic analysis for literature research. We also applied design science for artifact-driven development, specifically for the creation of an algorithm.

4.1 Systematic Literature Review

Our literature research involved the execution of a systematic literature review and qualitative data analysis. Both methods are described independently first before we present how we combined them. The first paper by Buchner and Riehle [14] details the SLR and provides insights generated through the literature review and subsequent thematic analysis. Building upon these insights, the subsequent article by Buchner and Riehle [13] introduces a research model.

The motivation behind the SLR and thematic analysis was to gain a comprehensive understanding of current research and industry perspectives in a structured manner. This approach facilitated the collection and analysis of qualitative data, serving as a holistic foundation for subsequent tool development work. As the methods were used in two research papers of this dissertation, the description follows the methodology description already outlined in two papers of Buchner and Riehle [14, 13].

4.1.1 Systematic Literature Review by Kitchenham

Goals and General Process: The primary objectives of Kitchenham's SLR guidelines [36] are not only to assist researchers in the identification and selection of relevant literature, but also to guide them through the subsequent data analysis and documentation processes.

Kitchenham divides the process of conducting a SLR into three phases:

1. Planning the review
2. Conducting the review
3. Reporting the review

Within each of these three phases, several steps are executed.

Steps and important aspects: The first phase conducted for the SLR involves planning the review before the actual search begins. The primary objective is to establish a clear foundation for the subsequent literature work. The following steps need to be conducted:

- Identifying the need for a review: Understanding the rationale and having well-defined objectives in mind is crucial when conducting the review. This step significantly influences the selection of literature and the resulting focus of the outcome.
- Specifying the research question: Part of the preparation includes clearly defining a research question that aligns with the overall review goal. The research question is an integral part of the review protocol.

- Developing a review protocol: The review protocol is formulated not only to prepare for the upcoming research but also to mitigate researcher bias by establishing all essential aspects in advance. The review protocol includes the research question, quality criteria, inclusion and exclusion criteria, and keywords for the literature search.
- Evaluating the review protocol: It is essential to evaluate the review protocol to ensure that the research adheres to the predefined objectives.

In the second phase (conducting the review), the following steps are executed:

- Identification of research: This steps involves identifying literature in the specified data sources using the keywords outlined in the review protocol.
- Study selection: The abstracts and content of the papers are scanned and analyzed to determine their alignment with the overall research goal and adherence to the inclusion criteria. Papers that do meet the exclusion criteria outlined in the review protocol are excluded.
- Study quality assessment: In this step, the remaining papers are re-evaluated to determine if they meet the quality criteria defined in the review protocol.
- Data extraction: Based on the gathered literature, relevant data for further research are extracted.
- Data synthesis: Finally, the data is collected and summarized for the report.

The last step is reporting the review, which, in our case, is accomplished by publishing the related research papers and this thesis. It is important to note that although the steps appear to be sequential, the overall literature review can be iterative in nature [36] as new insights from the data can help refine the research goals, research questions, keywords, and other aspects specified in the review protocol.

Review Protocol: As part of the methodology, it is crucial to present the key details of the prepared review protocol. Since the review protocol was already outlined in the related papers of this dissertation, only the most essential aspects are presented here. The extended version can be found in the appendices A and B.

Databases: For this review, the following databases were selected: ACM Digital Library, Google Scholar, IEEE Xplorer, Springer Link, Ebscohost, Wiley, and Scopus.

Keywords: This review concerns the business process domain (economics) and computational tools and techniques (computer science domain). Consequently, a set of domain-specific and common keywords were chosen to facilitate the search in both areas.

The common subset of keywords includes:

(Inner source OR open-source OR collaborative development OR cross-boundary collaboration OR cross border collaboration OR internal open-source OR software engineering OR software development OR DevOps OR agile OR platform)

For the business processes domain and general economic fundamentals related to inner source, the following keywords were selected:

(Business processes OR management OR accounting OR controlling OR taxation OR transfer pricing OR organization OR businesses OR enterprises OR organizational principles OR organization forms OR absorption costing OR cost calculation OR project management OR risk management OR product management)

Lastly, for the algorithmic and software development basics, the following keywords were applied:

(Software development OR programming OR ((cost OR effort) AND (calculation OR prediction OR estimation OR measuring OR quantifying OR computing OR calculating)) OR measurement OR KPI)

Quality Criteria: The selected papers needed to meet several quality criteria, including being peer-reviewed and published in a recognized journal or conference. Additionally, we considered technical reports. Algorithms and other computational tools and techniques had to be comprehensible and reproducible for our use case. Reports from reputable organizations such as the OECD, particularly in the case of transfer pricing, were also included. Furthermore, the rigor and relevance criteria introduced by Ivarsson and Gorscheck [32] were employed. Rigor was assessed based on the research context, overall study design, and validity of the analyzed paper. Regarding relevance, we ensured that the study context demonstrated the industry relevance of the content.

Inclusion and exclusion criteria: In our literature selection, we included papers that:

- Are related to inner source measurement in general
- Present tools or techniques for measuring or predicting business (process) related aspects
- Address problems within businesses and their processes connected to cross-boundary collaboration
- Calculate work effort or costs on different levels (code-level, project-level, or business-level)
- Provide noteworthy insights useful for measuring inner source and affected processes

We explicitly excluded papers that:

- Have no thematic connection or usability within inner source, cross-boundary collaboration in general, or related business processes
- Present tools and techniques that are non-repeatable, particularly affecting machine learning algorithms, mostly for effort estimation

- Present tools and techniques that are not adaptable to inner source development (not able to assess cross-boundary IP-flow)

4.1.2 Thematic Analysis by Braun & Clarke

Goals and general process: In addition to the SLR by Kitchenham, we applied thematic analysis by Braun and Clarke [10]. This method helps to identify and analyze patterns in data, enabling a detailed description of insights gathered from the dataset. In our case, we utilized the literature we previously identified.

The outcome of the thematic analysis is a list of patterns (themes) representing features of the dataset. The method organizes several key concepts (codes) into a theme, establishing a hierarchical structure of concepts and their dependencies.

Themes and codes are generated through iterative analysis of the literature, highlighting dependencies between individual concepts, often through the creation of a thematic map.

Steps: The thematic analysis guidelines by Braun and Clarke [10] consist of six basic steps:

- Phase 1) Getting familiar with the data: Initially, the researcher should become familiar with the data by reading it and noting down initial ideas and concepts.

- Phase 2) Generate initial codes: Following the initial reading, the first concepts (codes) across the entire dataset should be created and recorded.

- Phase 3) Create candidate themes: Based on the initial concepts, codes should be collected and analyzed to group similar concepts together, revealing their relationships (e.g., within a thematic map).

- Phase 4) Review themes: Using the initial concepts, themes should be reviewed, including the merging of similar themes or the splitting of ambiguous ones. This phase ensures that the identified codes and the original data are consistent with the themes.

- Phase 5) Define and name themes: Once themes and codes are consistent, clear names and distinct definitions should be assigned. This facilitates a common understanding among researchers working with the themes regarding the details of each code and theme.

- Phase 6) Produce a report: Finally, the themes and codes should be presented. Braun and Clarke [10] emphasize that the description should go beyond merely reporting the results, urging researchers to delve deeper and utilize the insights to answer the research question.

Braun and Clarke [10] emphasize that these steps are not rigid rules to be strictly followed linearly. Researchers should move back and forth, dynamically adjusting the codes and themes according to the most recent insights gathered during the coding process. We particularly utilized this approach.

4.1.3 Method Combination

We integrated the SLR methodology by Kitchenham [36] with the thematic analysis approach developed by Braun and Clarke [10] due to their complementary nature. Notably, Braun and Clarke did not explicitly define the data retrieval process, as their method is applicable to various forms of qualitative data, not exclusively literature. We employed the SLR guidelines by Kitchenham, which focus on the systematic identification and selection of literature. However, it is important to highlight that the SLR method, as outlined by Kitchenham, does not provide as detailed instructions for data analysis and synthesis as Braun and Clarke did in their thematic analysis.

Figure 4.1 illustrates the integration of both methods. The first two columns represent the three phases defined by Kitchenham. In our case, we replaced the data extraction and data synthesis steps with the thematic analysis approach, depicted in the third column.

The figure also emphasizes the iterative nature of both methods, which we used throughout our research. Specifically, we revisited the literature identification step after defining the initial

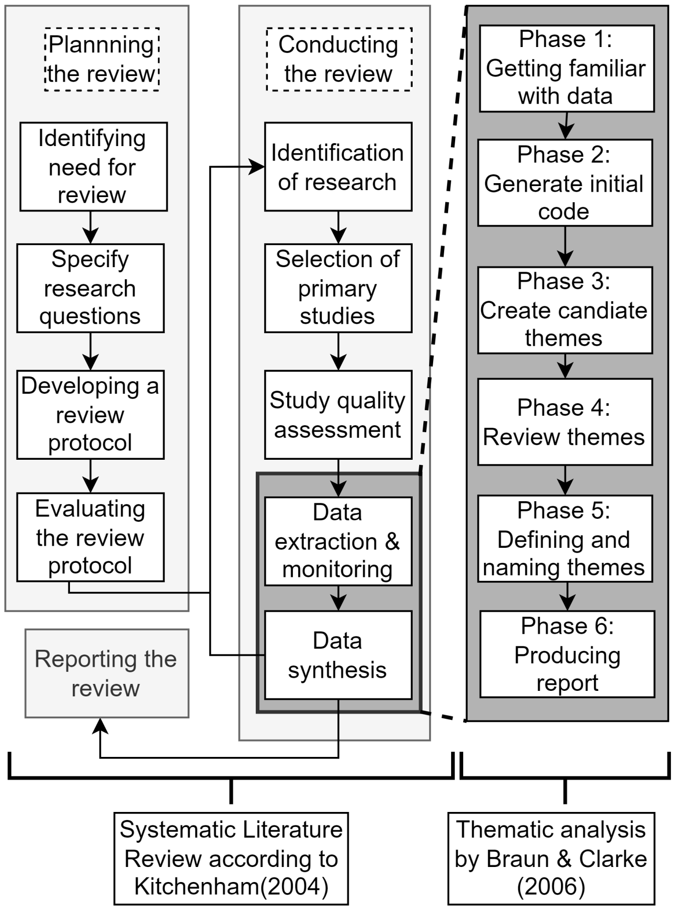

Figure 4.1: Overview of the SLR process based on Kitchenham [36] and Braun & Clarke [10], taken from Buchner and Riehle [14, 13]

set of themes and codes. The insights gained during the coding processes enabled us to refine and adjust the overall research process, facilitating the identification of literature more suited to addressing our research questions.

Hereafter, the term SLR refers to the combined method we applied, incorporating both the Kitchenham process and the thematic analysis approach by Braun and Clarke, rather than solely referring to the Kitchenham process.

4.2 Design Science

The second method we applied is design science by Peffers et al. [49]. The method was used for one article of this thesis, namely Buchner and Riehle [12]. As a consequence, the following methodology description is oriented on the content already described in that paper.

Goals: Design science is a framework for artifact-based research that solves real-world problems, especially in information system research. It follows an iterative approach aimed at developing various artifacts such as tools, methods, algorithms, models, or theories, providing a process for their development in research.

Steps: Design science, as proposed by Peffers et al. [49], outlines six basic steps:

Step 1: Problem Identification. In the initial phase, the problem is identified and justified, demonstrating the significance of the topic. Our motivation stems from interviews and extensive literature analysis, which are further elaborated in the paper (Buchner and Riehle [12]).

Step 2: Objective Definition. Building upon the identified problem, the objectives of the artifact are established, specifying the criteria the solution should meet. In our case, the objective is to develop an algorithm applicable for calculating software development costs within business processes, based on individual code contributions (commits).

Step 3: Solution Design & Development. Drawing from the established objectives, the solution solving the problem is designed. This involves the creation of the artifact, followed by a presentation of its functionality and architecture.

In our study, we developed an algorithm and implemented it in software. While Peffers et al. [49] proposed one step, we divided this step into two sub-steps: solution design and implementation. The solution design section explains the fundamental principles of the algorithm and its derivation from the dataset. The implementation section then shows which steps were taken to be able to execute the algorithm in software.

Step 4: Demonstration. This step shows the effectiveness of the solution in resolving the identified problem, often through simulations or by demonstrating the solution's functioning with specific inputs. In our case, the algorithm stands as the primary outcome. Therefore, the demonstration presents each case of the algorithm and its corresponding outputs, based on the example industry data set provided as input.

Step 5: Evaluation. The goal of the evaluation is to show that the defined objectives are fulfilled and the problem is actually solved. In our research, we used the industry data that already provided the basis for the solution design, development and demonstration. We conducted member-checking through interviews and made sure that the solution we designed actually solved their problems, validating it for the usage within cost calculation purposes.

Step 6: Communication. The final step involves the structured communication of not just the artifact itself but the entire design science process. This is achieved through the publication (Buchner and Riehle [12]) and this dissertation.

5

Results

This section presents the results of the research. We begin by discussing the fundamentals of inner source measurement and its impact on business processes, as explored in our literature review (Section 5.1). Next, we introduce an inner source research model (Section 5.2). Finally, we look into the design and implementation of an algorithm for calculating the cost of business processes in the context of inner source (Section 5.3).

This section aims to highlight the key insights from the three primary research articles contributing to the cumulative dissertation. The primary objective is to discuss the key findings from the original research, enabling a comprehensive presentation and subsequent discussion of the overall research results. Each of the three articles originally posed their dedicated research questions. To be able to distinguish those paper-specific research questions from the overarching research question of this thesis, the former are referred to as *P1-RQ*, *P2-RQ1*, *P2-RQ2*, and *P3-RQ*.

5.1 Systematic Literature Review

To address the central research question we conducted a systematic literature review and extracted insights applicable to our work. The following presents the key results that were previously published as:

[14] Stefan Buchner and Dirk Riehle. 2023. The Business Impact of Inner Source and How to Quantify It. ACM Comput. Surv. 56, 2, Article 47 (sep 2023), 27 pages. https://doi.org/10.1145/3611648

The original paper can be found in Appendix A.

5.1.1 Goals and Research Question

Our systematic literature review aimed to examine literature in the domains of business processes as well as computational tools and techniques for measuring software development efforts. Early in our research, we realized that the economic evaluation of inner source lies at the intersection of these two domains. Despite their distinct publishing avenues, keywords, and target audiences, our review incorporated insights from both areas, emphasizing the integration of the two. The review protocol described in the methodology section (Section 4.1.1) addresses this combination.

Consequently, the SLR utilized insights from both domains. On one hand, the review aimed to identify the impacts of cross-boundary collaboration in software development, particularly inner source, on businesses and their processes. On the other hand, we examined tools and techniques for measuring inner source or software development efforts in general. Based on these insights, we established connections between the two domains, including an analysis of how to measure the affected business processes with the identified tools and techniques. We outlined the essential properties that a tool or technique should possess to be applicable for inner source measurement. Furthermore, we classified the existing tools based on their usability for inner source measurement.

The research question derived from this motivation was as follows:

P1-RQ: What is the economic impact of inner source on companies and how can it be quantified?

We answered this question through an SLR following the guidelines of Kitchenham [36] and the thematic analysis of Braun and Clarke [10], as explained earlier. Our process involved three major iterations.

Overall, we identified 52 relevant papers across both domains, spanning back to 1984. Predominantly, articles from the economic domain were published earlier, while most articles describing computational tools and algorithms emerged within the last decade. Since this time, there has also been a growing interest in inner source.

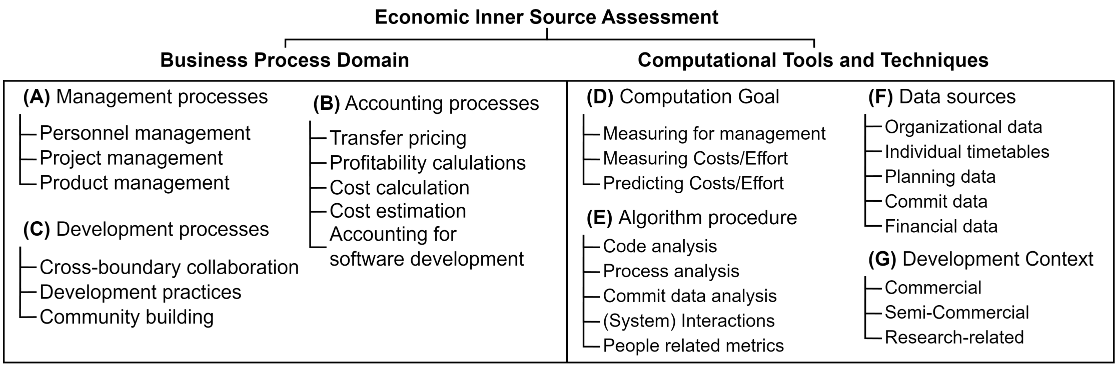

Figure 5.1: Overview of themes and codes, based on Buchner and Riehle[14]

In the end, we classified 27 codes across 7 themes from both domains. Figure 5.1 presents an overview of the themes and codes, with the left side displaying 3 themes and 11 codes related to the business process domain. On the right side, 4 themes and 16 codes related to the computational tools and techniques for measuring effort or inner source are presented.

It is crucial to note that although the research, especially the coding process, in both domains was conducted independently, some codes were based on the same literature, highlighting a connection between the two domains. The subsequent analysis looks further into this connection.

5.1.2 Business Impacts of Inner Source

In the domain of business processes, we identified three major sub-domains that are influenced by inner source measurements or cross-boundary collaboration in general: Management processes (Theme A), accounting processes (Theme B), and development processes (Theme C). While managerial and accounting processes are used to manage the company, development-related processes aim to develop the actual product.

The SLR revealed several managerial tasks and processes that are affected by inner source (Theme A). This includes project management tasks [33, 62, 31], such as project risk management [38, 25, 55], and related key performance indicators (KPIs) [19, 4]. These aspects are influenced by inner source as the paradigm shift involves different organizational units, particularly in the development of platforms [52]. This influence is also evident in product planning, as products are designed differently when platforms are involved, thereby affecting the entire product life cycle [24]. Additionally, personnel management approaches are impacted, as inner source deviates from the traditional principles of assigning developers to projects, allowing for more flexible assignments [15, 52].

Closely related to the managerial processes are those involving accounting (Theme B). One major challenge in the context of inner source arises from the fact that contributions within inner source span organizational boundaries. Calculating the value of such flows (e.g., for taxation) is referred to as transfer pricing, an established field [46, 50, 48, 41] that presents potential issues related to profit shifting [12, 46, 60]. Furthermore, inner source also affects how profits are allocated to the contributing organizations, a crucial aspect of operational business processes [58, 12]. This impacts cost calculation processes [6, 15, 12, 31] and various established cost prediction methods [8, 64, 35, 61]. Although initial accounting approaches addressing these issues exist [5, 31, 37], none of them comprehensively solve these problems in inner source.

In addition to the managerial and accounting processes, the development process itself also needs to adapt to inner source (Theme C). A core issue arises from the cross-boundary contributions made during inner source development [57, 56, 52, 15, 27], which do not align with traditional organizational models [29, 28]. However, traditional accounting and management processes adhere to these traditional organizational approaches (Conway's Law [20]) [58]. The SLR demonstrates that there is currently no ideal solution. Furthermore, new community building practices and processes must be established to fully leverage the advantages of inner source. This includes inner source incentivization schemes [17, 56, 21] and communities like the InnerSource Commons [2].

5.1.3 Tools and Techniques for Inner Source Assessment

The second review domain focused on computational tools and techniques used to assess the effort or cost involved in software development. The review did not contain all potential software effort estimation algorithms or principles. It aimed to identify and categorize basic concepts. The goal was to show which are suitable for the application within inner source and which not (see subsequent section). Each identified tool or technique possesses the capability to measure business or process-related aspects within inner source.

Overall, we classified four themes: Computational goals (Theme D), underlying algorithmic procedures (Theme E), used data sources (Theme F), and the overall development context (Theme G).

The tools and techniques we identified, independent of the coding in the business process domain, show diverse computational goals (Theme D). Specifically, we found dedicated tools and techniques tailored for measuring software development for managerial purposes [19, 4, 7, 30, 52]. Additionally, various tools and techniques are capable of retrospectively measuring the cost or effort of software development [34, 44, 5, 30, 54, 12]. Moreover, some focus on predicting future effort or costs [8, 64, 35, 51, 61, 9].

In the literature, various algorithmic procedures (Theme E) are described regarding how these goals were achieved. These procedures range from code analysis [5, 64] to commit data analysis [64, 12, 44, 54, 23, 4, 19, 61], people-related metrics [64, 44, 54, 51, 19, 7, 61], analysis of system interactions [64, 5, 30], and techniques that analyze the development process itself (e.g., benchmarking sprints) [64, 35, 23, 4, 19, 7]. For the inner source-specific use case, we needed to identify those procedures that are best suited for measuring cross-boundary collaboration or supporting the adjustment of existing business processes (See following Section 5.1.4).

The tools and techniques we reviewed utilize various data sources (Theme F). Commit data is the most prominent source for measuring software development [64, 44, 5, 54, 30, 23, 4, 19, 61, 12]. Additionally, planning data of development processes is commonly used [8, 34, 35, 51, 4, 19, 7, 61, 9]. Other sources include organizational data [51, 23, 19, 7, 61, 12], financial data [8, 5, 4, 19, 7, 12], and individuals' timetables [5, 23, 19, 7]. It is evident that a wide range of data sources can be useful for measuring and assessing software development within a company and can be instrumental for effort estimation.

Lastly, we identified the context in which the tools and techniques were developed (Theme G), although this aspect plays a minor role in determining the suitability of certain approaches for the inner source domain. Most of the articles we identified were developed in a research setting. However, we also reviewed literature that described commercially developed tools and techniques published in research outlets [8, 9]. Although non-publicly available tools and techniques developed in commercial environments are also accessible, they were not the focus of this research, as we aim for a publicly available, reproducible solution.

5.1.4 Thematic Dependencies

In addition to presenting the themes and codes, we also discuss the dependencies between them. These dependencies emerged during the coding process and are illustrated in the form of thematic maps. The thematic maps are providing valuable insights into the interconnection between the two domains and how inner source and associated processes can be quantified.

Two primary connections of interest were identified:

1. The dependency between the business processes affected by inner source (Theme A to C) and the goals of the computational tools and techniques (Theme D).

2. Classification of the suitability of algorithmic approaches (Theme E) for various computation goals (Theme D).

In the first case, the detailed connections can be observed in Figure 5.2.

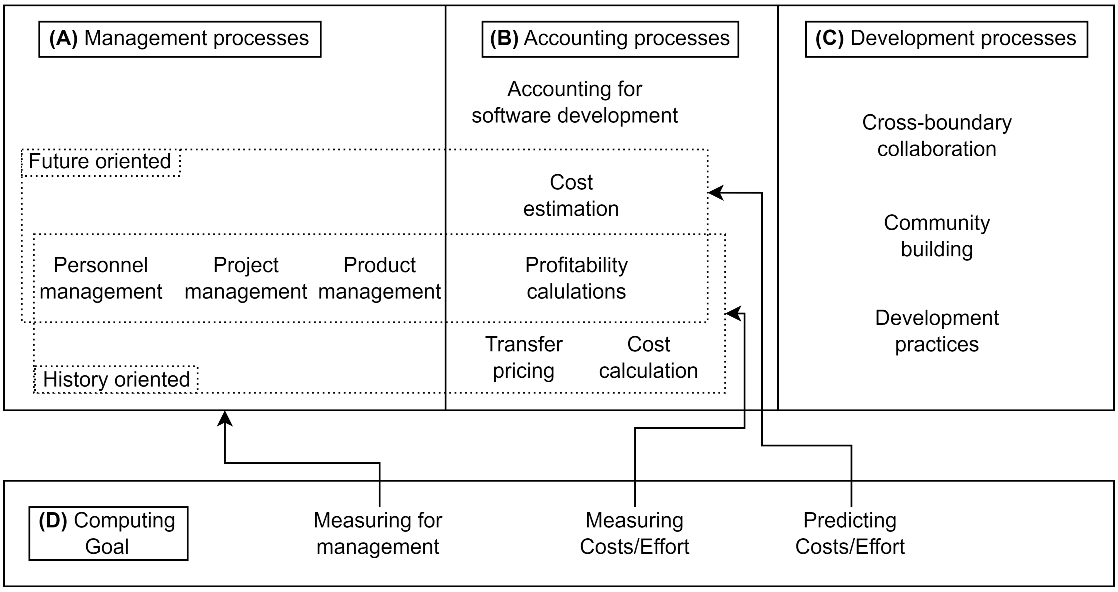

Figure 5.2: Thematic map of the business process view and tools and techniques, adapted from Buchner and Riehle [14]

During our research, we observed that the independent review domains had overlapping contents/codes, although they operate on different levels. The reviewed tools and techniques were primarily technical, while the business processes involved functional tasks within a company. Nevertheless, they were connected from a content perspective.

Some classified tools and techniques (of Theme D) were predominantly designed for measuring management-related tasks and processes. Those align with the management process theme of the business process review domain (Theme A). This relationship is depicted in Figure 5.2 through the arrow linking "measuring for management" in Theme D and the management processes (Theme A).

Furthermore, we identified that some of the analyzed business processes focused on historical data and retrospective calculations. One example are the calculation of transfer prices for taxation. Other processes were more predictive, like cost estimation. Certain business processes, such as product management or project management, fell somewhere in between. That indicates that the historical/future-oriented differentiation is largely independent of the specific domain, even though management tasks tend to be more future-oriented.

A similar differentiation was also noted in the independent coding of the computational tools and techniques. We classified the codes "Measuring Cost/Effort" and "Predicting Cost/-Effort," corresponding to the historically oriented and future-oriented business processes, respectively.

The second set of thematic dependencies between the codes are depicted in Figure 5.3. This figure shows the connection between the algorithmic procedures and the computational goals.

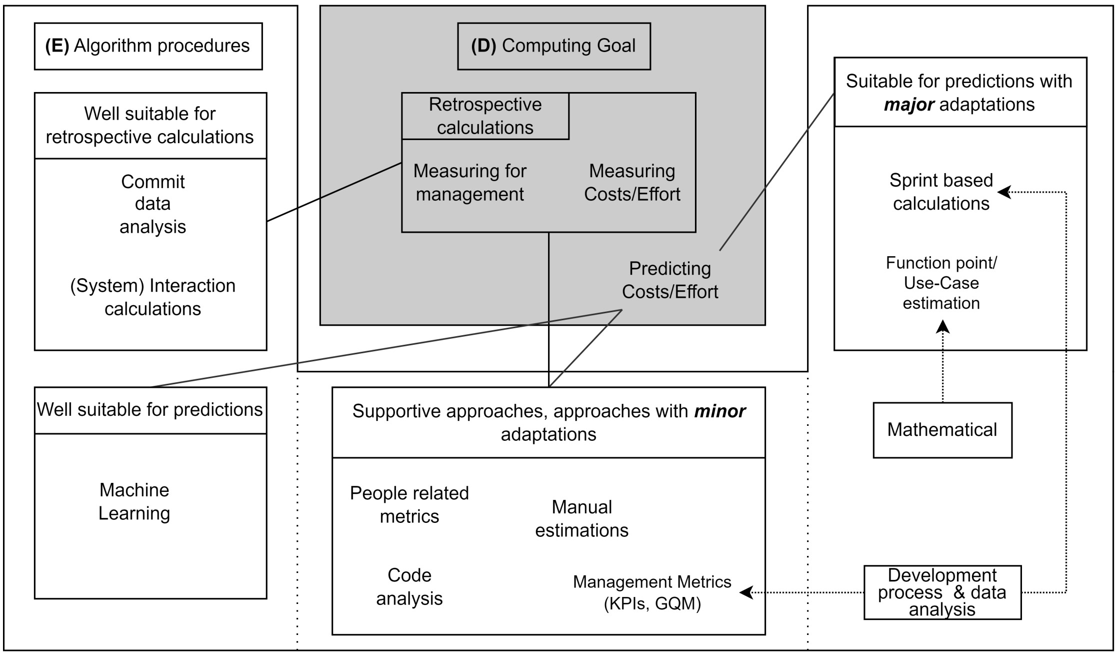

Figure 5.3: Usability of algorithmic procedures for computational goals, adapted from Buchner and Riehle [14]

This analysis reveals the extent to which algorithmic procedures can be used for various calculations related to inner source measurement. Notably, utilizing system interactions and commit data proved most suitable for inner source metrics, especially for retrospective calculations. Reason is that these type of daily tasks can easiest be bound to IP flow crossing organizational boundaries. Machine learning approaches were found to be best suited for predicting inner source collaboration, although this wasn't the primary focus of the review.

The majority of the identified approaches (e.g., source code analysis) were found to be suitable as supporting metrics or requiring minor adaptations to measure cross-boundary collaboration. In contrast, some approaches need major adaptations, such as sprint-based calculations or function point/use-case estimations. These are not easily applicable to single IP contributions across organizational boundaries.

The key takeaway is that while some algorithmic procedures are well-suited for the inner source domain, the majority require either minor or major adaptations before they can be used effectively. As a result, this analysis serves as a guideline for future research, supporting in the creation of metrics usable for inner source measurement.

Another interesting insight is visible when considering all the thematic connections together. The first insight (Figure 5.2) illustrates the connections between the business processes influenced by inner source (Theme A to C) and the computational goals (Theme D) of the identified tools and techniques. The second main set of connections (Figure 5.3) exists between the computing goals (Theme D) and the algorithmic principles (Theme E). Consequently, that shows how applicable the identified algorithmic principles in Theme E are for solving challenges related to the business processes of Themes A to C. Thus, those principles can contribute to answering the overarching research question on inner source quantification.

In conclusion, the principles we identified can and should be utilized in subsequent research to facilitate the adaptation of businesses to inner source. This also addresses the research question P1-RQ posed in this underlying paper.

5.2 Inner Source Research Model

In this section, we present the inner source research model developed as a result of our literature review on inner source quantification. The insights shared in this section were originally published as:

[13] Stefan Buchner and Dirk Riehle. 2023. A Research Model for the Economic Assessment of Inner Source Software Development. In Proceedings of the 56th Hawaii International Conference on System Sciences, HICSS '23. ScholarSpace, Maui, Hawaii, 353–364. https://hdl.handle.net/10125/102672

The original paper can be found in Appendix B.

5.2.1 Goals and Research Questions

The primary objective of this research was to create a research model focusing on inner source measurement. This research model was designed to illustrate the relationships between the computational aspects and the business impact of inner source. In contrast to our literature review and thematic analysis, the research model concentrates solely on the research perspective rather than practical, applicable algorithms and tools. It outlines how these topics interconnect and provides guidance for future research in the field of inner source measurement.

The research model addresses the following research questions:

P2-RQ1: What is the current state of research in economic inner source assessment?

P2-RQ2: What are current challenges of economically assessing inner source and how can they be tackled?

The first research question P2-RQ1 follows to the fundamentals already addressed and described in the SLR. The second research question P2-RQ2 looks into the challenges that must be addressed to enhance inner source measurement, as presented within the research model. It's important to note that the themes and codes related to the business process perspective in the SLR, as initially presented in the research model paper (Buchner and Riehle [13], Appendix B), differ from the final version outlined in the SLR paper (Appendix A) and this thesis (Section 5.1). The reason are long review times between which additional reviews improved the final version of the themes and codes. However, these changes do not impact the research model's foundation, which is built on insights obtained from the SLR. Consequently, the themes and codes are not presented again.

The SLR illustrates how inner source influences various business processes and identifies tools and techniques suitable for quantifying these effects. In summary, the SLR answers *which problems* can be addressed and *which basic principles* should be applied. However, it falls short of detailing *how to progress* in future research and lacks a unified framework for understanding future inner source measurement research.

5.2.2 Research Model Layout

The resulting inner source research model is depicted in Figure 5.4.

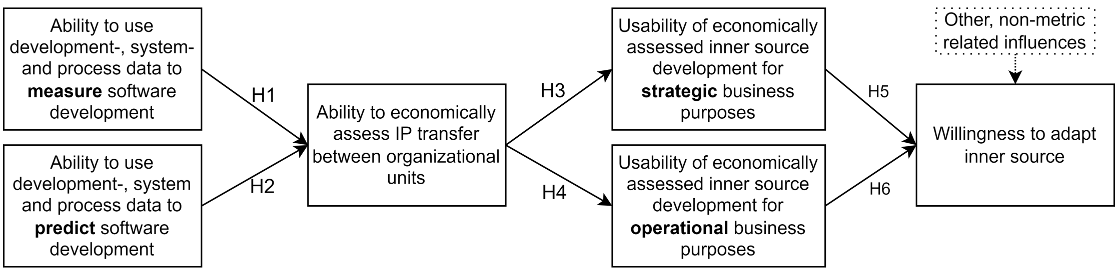

Figure 5.4: Inner source research model from Buchner and Riehle [13]

The research model is structured as a bottom-up approach, progressing from left to right, presenting a research path. The stages on the left show the foundational aspects that researchers need to work on and understand. These foundational aspects are necessary to reach a stage where inner source can be quantified and applied in business processes, as represented by the middle stages. Once these are in place, the overall adoption of inner source can increase through improved measurement.

Furthermore, the research model is built upon the different aspects identified in the thematic map of the SLR. It distinguishes between predictive and retrospective calculations from a computational perspective. Future research should include both predictive and retrospective metrics and tools, as these approaches differ algorithmically and target distinct use cases. However, both types of algorithms should ultimately be integrated into a unified toolset. These serve as the foundation for assessing inner source's impact on business processes.

From a managerial perspective, the research model distinguishes between strategic and operational business process measurement. Future research in tools and metrics should address strategic and operational processes. Due to the long-term nature of strategic business processes, predictive methods may be more relevant there. In future work, this could involve predictive data e.g. to support product planning decisions. In contrast, operational business processes typically rely on retrospective data e.g. to estimate the tax burden of individual legal units. The subsequent section (Section 5.3) presents an algorithm designed for such use cases.

In summary, the research model is structured into four distinct stages that form the overarching research path. Figure 5.4 visualizes this path from left to right. The first stage introduces the low-level computational perspective with the historical and predictive differentiation. The ability to perform both types of measurement using various data sources (development, system, and process data) is a prerequisite for assessing IP transfers in inner source (second stage). Financially assessed inner source contributions, in turn, serve as the basis for adapting strategic or operational business processes to the inner source paradigm (third stage). Finally, enhanced business integration of inner source facilitates overall adoption (fourth stage), enabling companies to realize the full benefits of inner source.

5.2.3 Impact on Future Research

Based on the dependencies between the different stages and the computational as well as managerial view of the research model, we derived six research hypotheses.

Hypotheses 1 and 2: *The ability to use development-, system, and process data to* ***measure/predict*** *software development correlates positively with the ability to economically assess IP transfer between organizational units.*

The first two hypotheses depict the connection between the initial stages, emphasizing the significance of measuring (Hypothesis 1) and predicting (Hypothesis 2) software development in inner source as the economic measurement basis.

Hypotheses 3 and 4: *The ability to economically assess IP transfer between organizational units correlates positively with the usability of economically assessed inner source development for* ***strategic/operational*** *business purposes.*

Following the pattern of the first two hypotheses, the subsequent two highlight the assumption that the ability to economically assess inner source serves as the foundation for various strategic (Hypothesis 3) and operational (Hypothesis 4) business processes within inner source.

Hypotheses 5 and 6: *The usability of economically assessed inner source development for* ***strategic/operational*** *business purposes correlates positively with the willingness to adapt inner source.*

The last two hypotheses anticipate the impact of strategically (Hypothesis 5) or operationally (Hypothesis 6) assessed business processes on the overall inner source adoption.

Research implications: The derived hypotheses guide the research progress when conducting measurement-related inner source research. They also include the challenges that need to be addressed, thereby answering the earlier proposed P2-RQ2. Additionally, this complements prior research on the current state of inner source, as conducted by Edison et al. [27]. While their work extensively discusses the definitions, benefits, and challenges of inner source, it did not look deeply into measurements and metrics.

Moreover, the research model lays the groundwork for addressing a significant challenge faced by companies when implementing inner source. Many companies lack the buy-in of middle managers as they fear losing their personal performance goals [3, 52]. By providing a clear path to link accurately measured contributions to different business processes and perspectives, the inner source research model has the potential to offer critical metrics for middle managers in their daily decision-making and performance assessments.

The research model provides the overall model on how to progress and related hypotheses, but it does not offer the solutions to the hypotheses. It is the task of future research to validate these hypotheses, for example, by creating measurement and prediction algorithms and tools or by integrating these measurements into business practices. The subsequent research presented in this thesis partially implements initial aspects of the research model. Additionally, future research should focus on developing associated guidelines or handbooks for working with measured inner source artifacts in practice.

5.3 Work Time Estimation Algorithm for Cost Calculation

This section focuses on presenting an algorithm for solving challenges highlighted in the previous sections. Specifically, it introduces an algorithm for estimating work time, intended for the purpose of calculating costs in inner source business processes. The algorithm was previously published as:

[12] Stefan Buchner and Dirk Riehle. 2022. Calculating the Costs of Inner Source Collaboration by Computing the Time Worked. In Proceedings of the 55th Hawaii International Conference on System Sciences, HICSS '22., virtual, 7466–7475. https://doi.org/10.24251/HICSS.2022.896

Therefore, the section presented here highlights results already thoroughly discussed in the original paper, which is also accessible in Appendix C. In addition, preparatory research for the article has already been published in the author's master's thesis [11].

5.3.1 Goals and Research Questions

The primary objective of this paper was to enhance cost-based business processes associated with collaboration across boundaries. This was achieved by proposing an algorithm that leverages the commit history of projects to compute the proportion of work time each organizational unit invests in individual projects. In inner source environments, each project typically has one primary responsible organization (the owner) and various contributors from different legal entities. The estimation of work time for each project can subsequently be used to compute the cost of individual contributions. Adhering to this logic, the design, implementation, and presentation of the algorithm followed a two-step approach:

1. Creating a work time estimation algorithm.
2. Utilizing it for cost calculation.

Hence, the main research question that the algorithm addresses is as follows:

P3-RQ: How can we calculate the time spend on code contributions for usage within various cost related business processes

A key motivator for the work time estimation algorithm is the taxation challenge as described earlier. The calculation of transfer prices pose a significant challenge in the inner source domain, as the article presents in more detail and we already described above. The presented algorithm was designed to address this problem. The result can be utilized to calculate the cost that one organizational unit must pay another.

The primary focus is on showing the algorithm's design, implementation, and overall architecture to facilitate cost calculations in inner source. The aim was not to provide an exact estimation for each individual commit (e.g., down to the minute) but rather to ensure accuracy in capturing the overall share across all projects.

The algorithm was developed as design-science artifact following the process by Peffers et al. [49]. The design science steps we followed were described earlier in Section 4.2.

Our development was based on real-world industry data from a multinational corporation. The data set included contributions from approximately 400 developers in 94 organizational units across four hierarchical levels. We analyzed approximately 230,000 file changes in about 29,000 commits from a total of 13 projects developed over a 1.5-year period. In detail, we investigated commits, organizational data, and individual developer data (such as name or organizational unit). Based on this, we conducted statistical analyses of development behavior, which are detailed subsequently. The insights from the analysis were then used to examine individual code contributions and estimate the time spent on each commit. The overall process was iterative which enabled us to improve the accuracy of the results over time. To ensure sufficient accuracy, we employed member checking to determine the cutoff point.

5.3.2 Software Development Statistics

Of the analyses we conducted, two particular statistics significantly influenced the resulting algorithm.

On one hand, we analyzed the distribution of commit timestamps (local commit time) throughout the day. This analysis helped in identifying peak commit periods. The graphical representation of this analysis can be found in Figure 5.5.

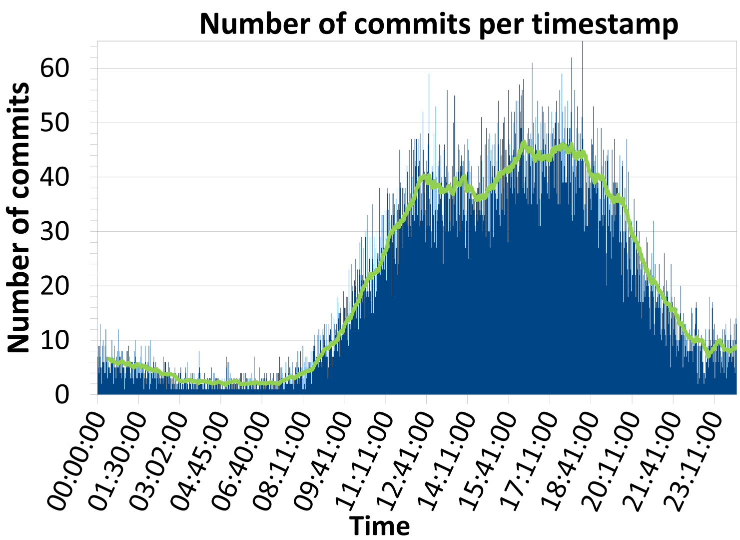

Figure 5.5: Number of commits per timestamp, taken from Buchner and Riehle [12]

Although the results may seem straightforward, their implications for the design of work time estimations are significant. The graph depicts the number of commits for specific timestamps. The number of commits rises in the morning, experiences a slight dip during midday, and gradually declines towards the evening. Notably, the absence of fixed work hours in modern work environments implies a lack of fixed patterns outside the general daytime distribution. Therefore, a work time estimation algorithm must adapt to commits made beyond regular working hours. That includes commits during short nights (e.g., last commit at 11 pm and the first commit the next day at 6:30 am), long workdays, and all scenarios in between. Consequently, an algorithm cannot solely rely on fixed work time patterns. However, understanding the commit distribution across all contributors helps us estimate typical development patterns.

On the other hand, we analyzed the time difference between two consecutive commits by the same developer. The results are illustrated in Figure 5.6.

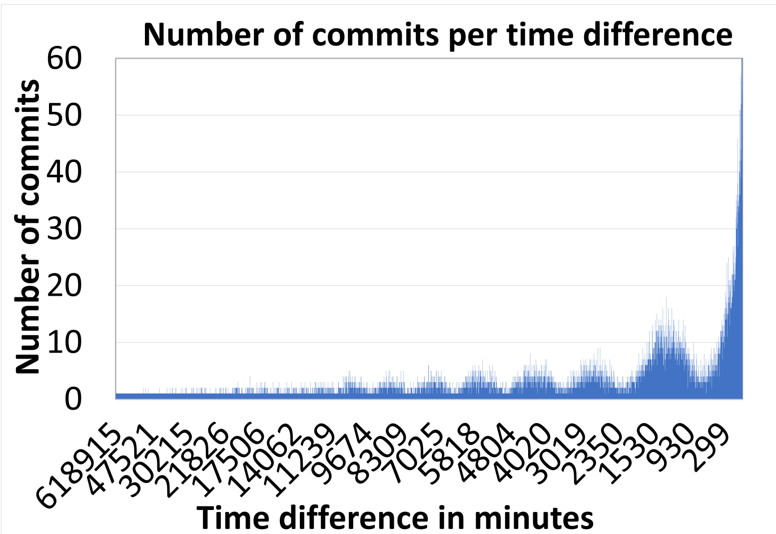

Figure 5.6: Number of commits per time difference

The findings suggest that most code commits occur in quick succession, often within minutes. The majority of commits take place within a 12-hour window (720 minutes), indicating that developers commit there at least twice a day. The analysis also revealed a spike in commits after 24 hours (1440 minutes) from developers who commit daily. The same trend recurs for several days (two to four days), even though with a lower total number of commits. Commits separated by more than a day usually align with weekends, holidays, or vacation periods.

5.3.3 Algorithm Design

The fundamental algorithm design is built upon the statistical analysis. The algorithm accounts for the time of day when commits are made while acknowledging that committing at night is less common. It makes use of the statistical pattern and accepts inaccuracy in the estimation of individual commits in favor of broad-scale accuracy.

The algorithm examines the commits of a single developer. It estimates the time spent on each individual contribution by considering the scope and size of the commit, the time elapsed since the previous one, and other available metadata. The algorithm reconstructs a developer's workday by comparing the commit properties to those of the large-scale statistics. Notably, the algorithm exclusively focuses on commits, disregarding all non-coding activities. This deliberate choice simplifies the algorithm, allowing for potential expansion in future research. As said before, the primary objective of the algorithm is to estimate the proportion of work time that an entire team or organization invests in individual projects.

As revealed by our statistical analysis, the majority of commits were made at least once per day, representing the *regular cases* that the algorithm handles. Additionally, the algorithm estimates the time spent on commits where no previous commit is available (e.g., the first commit overall) and commits made after several days (e.g., weekends, holidays). Those are referred to as *irregular cases*. In these scenarios, the estimation of work time needs to rely solely on the data provided by the commit, such as changed lines and timestamps, without reference to the overall time elapsed since a previous commit.

While the detailed equations describing the algorithm's mathematical aspects are available in the original paper (Buchner and Riehle [12] and Appendix C), this section primarily provides a narrative description of each case of the algorithm to simplify the presentation.

Regular cases: Commits that are closely related to previous ones in terms of time difference are classified as regular cases. Specifically, this includes commits made by a developer within 36 hours after the previous one. This 36-hour threshold reflects the daily commit behavior as observed in the statistical analysis (the dip at around 2160 minutes in Figure 5.6). Within this time frame, the following cases (derived from the statistical analysis) are define.

Time difference < 6 Hours: This case is based on the analysis suggesting that the majority of commits are made within minutes. Given the close temporal relationship to the previous commit, it is assumed that the time was spent on development. Therefore, the commit is assigned the full time difference as working time. This method is sufficient enough when it comes to calculating the work time share spent per project.

Time difference 12 to <36 Hours: The second case includes commits made between 12 and 36 hours after the previous one. Unlike commits made a few minutes apart, there cannot be an assumption that the entire time was spent on development. These cases typically represent daily commits. In such cases, it is highly probable that at least one night elapsed. To estimate the work time of these commits, the number of historically made commits in the time span since the last one are compared to the number of overall commits. The basis for this comparison is the commit distribution illustrated in Figure 5.5.

For example, if a commit was made at 8 am and the previous one at 5 pm on the day before, the time difference is 15 hours. As most of this time interval falls during the night, it is highly likely (compared to other commits) that minimal to no work was done during the night. However, this cannot be guaranteed. The estimation is calculated by comparing the number of historical commits between 5 pm and 8 am with the overall number of commits. If, for instance, 10,000 out of a total of 150,000 commits were made during this time frame, it is considered to represent 6.6% ($10,000/150,000 = 6.6\%$) of the typical work conducted during such periods. Hence, the commit receives 6.6% of the typical work time of a developer during a day.

The approach dynamically adjusts the resulting estimation based on the number of historical commits within a specific time frame. It enables a more nuanced allocation of work time depending on the time of day the commit was made. Consequently, commits made 24 hours after the previous one are allocated a full work day, whereas those further apart receive more, and those closer receive less work time. The length of a typical work day can be input by the implementer of the algorithm, either as a fixed value for all developers or dynamically adjusted for each developer based on real data (e.g., input from HR systems).

Time difference 6 to < 12 Hours: The third regular case represents all commits made six to twelve hours after the previous one. This may for example occur after a short night or during regular to extended work days. After examining the data set, no clear differentiation between these variations was feasible. To take this into account, the night time is considered in this case. The night time period for the algorithm can be either fixed or dynamically determined based on the number of commits during a specific time frame. It is also possible to base it on developers individual behavior.

The time span of the commit classified as daytime receives full allocation as work time, identical to the approach described above (case < 6 hours time difference). The time span classified as nighttime is allocated proportionally based on the historical share, as seen in the 12 to 36 hours case. This mixed case allows for the inclusion of both extended regular work days and nighttime hours.

Irregular cases: The irregular cases include all commits with no time difference or those made after more than one day elapsed (a time difference exceeding 36 hours). In our statistical analysis, we found a correlation between the lines of code (LoC) a commit affects (adds, changes, deletes) and the resulting work time in regular cases. The algorithm utilizes this statistical relationship to map the results of the regular estimation, where more contextual information about the work day is available, to the commits where such information is unavailable. Detailed results of the conducted analysis (linear regression) can be found in the paper [12]

(Appendix C).

It is essential to note that the use of LoC in combination with the results of the regular cases is just one example of how the irregular cases can be calculated. The algorithm is flexible enough to incorporate other input for this calculation. We chose LoC combined with the results of the regular cases to provide a self-contained solution that requires no external input and can be automated without manual intervention. An alternative approach could involve manual work time measurements where developers record the time spent. However, we did not pursue this approach as it is not feasible for high-frequency large-scale development, such as inner source.

5.3.4 Algorithm Application and Evaluation

The algorithm's output is the estimated work time for each commit. The initial objective wasn't to perform work time estimations, the aim was to conduct cost calculations. Consequently, each commit is tagged with author information, including the originating organization (the committer's organization) and the recipient organization (the project owner). This allows the identification of commits that cross organizational boundaries. Afterwards, work time is accumulated across the organizational hierarchy (team-level, then organization level and so on).

The results are then being used to calculate the share of work time for each organizational entity. That enables us to determine for each entity the proportion of work time dedicated to individual projects and their origin. For instance, if the results indicate that 15% of Team A's effort is directed toward a project managed by Team B, it implies that 15% of the overall costs are allocated to developing projects beyond the original scope of the department. If Teams A and B operate within distinct legal entities, the estimation also sheds light on the (cost-based) value of the IP flowing across organizational boundaries. This becomes crucial for calculating the tax burden of individual departments.

The algorithm is an important contribution for solving the transfer pricing challenge in inner source. It enables companies to conduct and implement inner source more securely, easing concerns about potential profit shifting.

To demonstrate the algorithm's efficacy, we utilized industry data. More specifically, we used the previously explained data set on which the statistical analysis were applied. The article (Buchner and Riehle [12], Appendix C) shows anonymized example JSON-outputs of the algorithm in more detail.

To evaluate our approach we conducted several interviews. Primarily, we ensured that the resulting work time share was sufficiently accurate for our purpose, achieved through member checking with the data owner. Furthermore, we verified the usability of the results for cost calculation, particularly within the crucial context of transfer pricing, through discussions with the German Ministry of Finance. The Ministry confirmed the suitability of our approach for cost calculation (as per the cost plus approach of the OECD [47]), highlighting its significant advantage over existing solutions relying on manual estimation.

However, the Ministry also emphasized that the solution might not be applicable to other transfer pricing methods that are not cost-based. Another key insight from the evaluation process was that, as expected, the data owner confirmed that the absolute estimation of work time is not accurate enough for purposes beyond calculating the share of time spent on individual projects. Nonetheless, given that the algorithm's original aim was to enable cost calculation, these results fall well within the algorithm's intended scope.

In conclusion, the presented algorithm facilitates cost calculation when inner source is applied. It not only enables the computation of cost-based transfer prices for taxation purposes but also lays the foundation for a wide range of cost-based metrics and processes in the domain of inner source.

6

Discussion

In the previous section, we explained the results from the individual contributing papers. While they were presented in a logical sequence, not all dependencies and thematic connections were explicitly highlighted.

Explaining these dependencies, the discussion serves a dual purpose. First, it looks into the relationships among the papers as a whole and the interconnections between various aspects within those articles. We provide detailed insights into how the individual results depend on each other. This deepens the understanding of the topic.

Secondly, the discussion goes beyond the content of the presented articles. We explore how outside research and unpublished work (as main and co-author), are influenced by the insights presented in this dissertation. We emphasize the importance of the SLR, particularly the thematic map, and the research model in shaping future research directions.

Figure 6.1 offers an overview of the thematic dependencies and illustrated how they build on each other. All the parts that were described in Section 5 are visualized in black. These connections are labeled as A to D for easier reference and comprehension. Additional insights

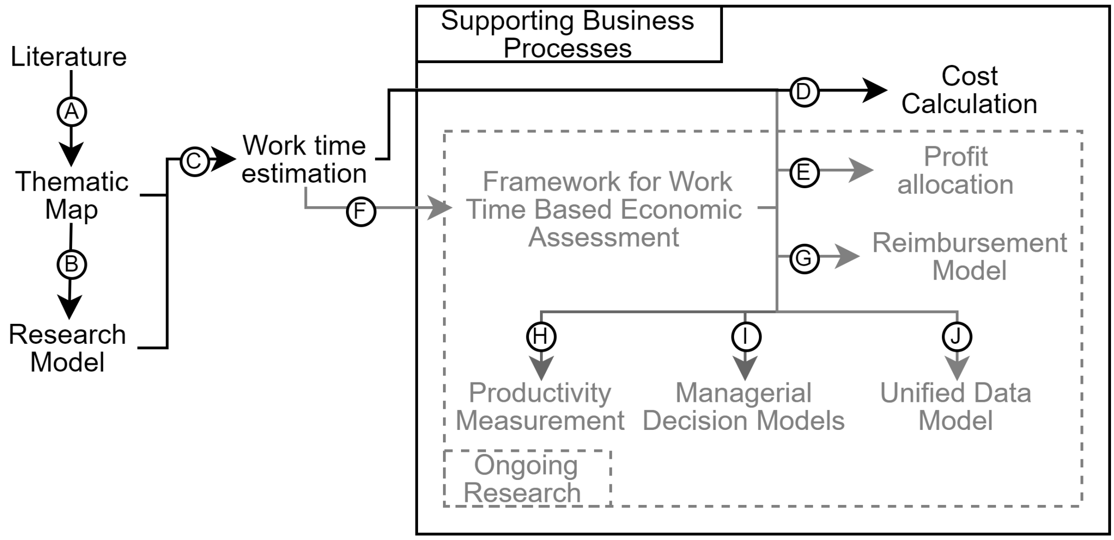

Figure 6.1: Overview of thematic dependencies

are depicted in a lighter shade, labeled as connections E to J.

Furthermore, the figure shows when the results can contribute to the actual improvement of business processes in the context of inner source, which is the overarching goal of this thesis. This contribution is encapsulated within the box titled "Supporting Business Processes". For instance, it includes the application of cost calculations based on the work time estimation algorithm. We discuss these detailed connections in the following sections.

It's important to note that these thematic dependencies are presented from a research perspective, with the literature serving as the foundational input. Additional inputs, such as industry data for the work time estimation algorithm, insights from conducted interviews and ongoing research are not visualized here to maintain clarity and ease of understanding.

6.1 Literature and Thematic Map Dependencies

The first significant dependency lies between the literature used as the basis for the SLR and the resulting thematic map (Connection A of Figure 6.1). These were extensively discussed in Section 5.1. A key aspect here is how these fundamentals align with the core research question of this thesis about quantifying inner source for business process usage. By building subsequent research on different review domains we ensured a solid, literature-based foundation.

Analyzing the thematic dependencies between the two review domains is crucial. The study identifies which types of algorithmic procedures are well suited for measuring inner source and how these approaches influence various business processes. These insights contribute to addressing the research question from a literature-based standpoint, thus laying the theoretical groundwork.

6.2 Thematic Map and Research Model Dependencies

Secondly, there are dependencies between the thematic map of the SLR and the research model (Connection B). As mentioned earlier, the research model is built upon the insights from the thematic map. However, specific details of this connection were omitted in the results presentation to maintain focus on the description of the research model. The research model stems from two main aspects of the thematic map.

The initial stages of the research model (Hypotheses 1 and 2) highlight the ability to measure and predict inner source. These two aspects were also covered in the presented thematic maps. The first thematic map demonstrates that some business processes emphasize predictive tasks, while others involve retrospective calculations (See Figure 5.2). Both types of processes are integral to managerial and accounting-related work. The second thematic map is the algorithmic suitability analysis conducted during the SLR (See Figure 5.3). It indicates that approaches based on individual IP transactions (e.g., commits, system interactions) are well-suited for measuring inner source. Consequently, the use of these approaches helps in addressing the initial hypotheses. They form the foundation for all future research and the development of practically relevant tools.

The subsequent hypotheses 3 and 4 of the research model assert that with the ability to assess IP transfer between organizational units, the data can be used for both strategic and operational business purposes. These assertions align with the findings of the thematic analysis, where the managerial processes influenced by inner source lean towards strategic implications, while the accounting processes tend to be more operational.

The study demonstrates not only that the hypotheses of the research model are derived from the thematic map, but also how the identified tools and techniques can contribute to enhancing inner source adoption by quantifying inner source business processes.

6.3 Thematic Map and Work Time Algorithm Dependencies

The work time estimation algorithm was presented without a clear explanation of its relationship to the thematic map or the research model. Although the thematic map primarily presents theoretical work and dependencies, it provides crucial inputs for the practical algorithm. The fundamental principles driving the algorithm from a technical perspective were derived from the thematic map. This helps to clarify the business process perspective, specifically the cost calculation use case, and facilitates planning for future research based on the algorithm's outcomes.

From a technical standpoint, the work time estimation algorithm utilizes principles outlined in the thematic map (Connection C). The algorithm is built upon the "well-suited approaches" highlighted in the thematic map (See Figure 5.3), particularly leveraging commit data as the foundation. Utilizing commit data is the simplest and most fundamental way of measuring inner source collaboration. Although the resulting work time estimation might not be precise for individual commits, it is robust enough to be employed in various business processes in the form of work time share per project. The algorithm design effectively bridges the gap between theoretical insights and practical application, thereby presenting an initial algorithm that can be employed to financially assess inner source development. Other approaches were classified in our literature analysis as suitable/suitable with minor adaptions. Those are not currently integrated into the proposed algorithm. Future research, particularly incorporating system interactions or comprehensive handling of full- and part-time contributions [54], may be promising for further exploration.

From a business perspective, the algorithm plays an important role as a decision-making tool in various business processes that are influenced by inner source development. Notably, the application for cost calculation (Connection D), specifically the demonstrated transfer pricing calculation, benefits from the algorithm we designed. This explicit connection (visualized in Figure 6.1) between the algorithm and the theoretical basics of the SLR demonstrates that the algorithm is not independent of the literature basics; rather, it builds upon and extends the theoretical foundations for practical use by practitioners.

In addition to the cost calculation use-case, the algorithm might also find applicability in other business processes influenced by inner source (Connection E). While the algorithm was primarily demonstrated and evaluated in the context of cost calculation, its application in scenarios such as internal profit allocation to individual departments could also be feasible and straightforward. This possibility arises from the algorithm's core objective, which is to estimate the work time share for individual organizational entities, thereby making it adaptable to the internal distribution of revenues among contributing departments. This flexibility could potentially address the previously mentioned concern of middle managers regarding the achievement of their performance goals [52]. Further research is needed to look deeper into this topic.

6.4 Research Model and Work Time Algorithm Dependencies

In addition to the thematic map, the work time algorithm is also linked to the research model (Connection C). While we demonstrated the algorithm with the cost-calculation use-case, we also recognized its potential applicability to other business-related inner source domains.

All of these instances are examples of measuring inner source. Consequently, the algorithm forms the basis for addressing Hypothesis 1 of the research model. The actual work time estimation covers the aspect of the first hypothesis that fits to the *ability to use development-, system-, and process data to* ***measure*** *software development* (first box in Figure 5.4). Moreover, the application of the estimated results to a business use-case (in our case, cost calculation) corresponds to the aspect of the first hypothesis related to the *ability to economically assess IP transfer between organizational units*.

Thus, we can conclude not only that the algorithm is rooted in the insights collected from the literature but also that it contributes to answering the first hypothesis of the research model. However, further measurements of inner source activities and broader applications in a wide variety of business processes must be undertaken in future research to justify the hypothesis.

6.5 Embedding of Ongoing Research

To understand the full scope of the research outlined in this dissertation an examination is required that extends beyond the articles presented here. The theoretical papers in this thesis lay the groundwork for an first implementation of an inner source assessment approach through the use of the work time algorithm. Furthermore, the contents discussed here can serve as valuable input or guidance for research beyond the findings of this dissertation.

Figure 6.2 provides an overview of ongoing and outside research and illustrates how they align with the research model.

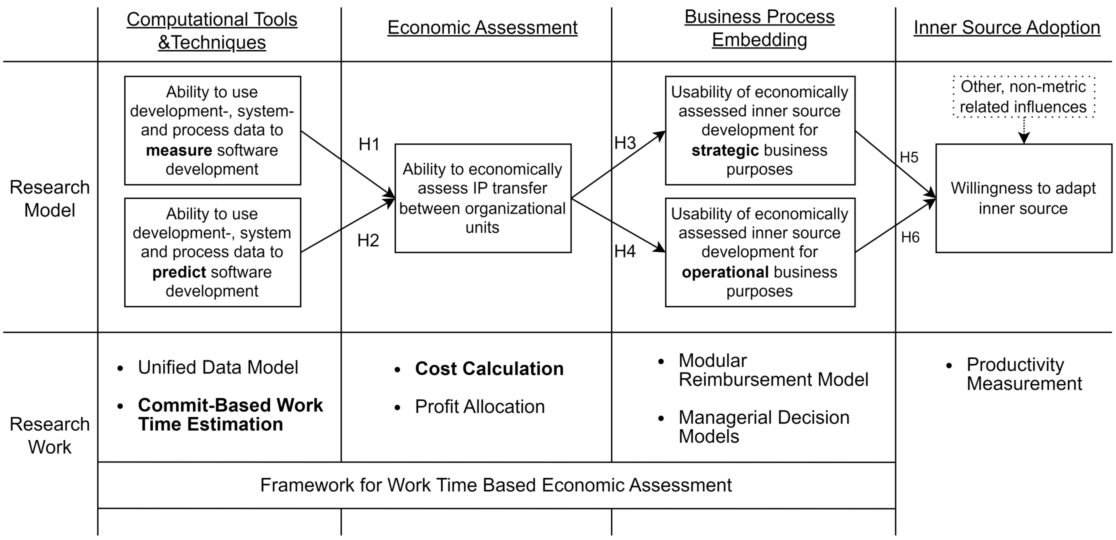

Figure 6.2: Alignment of outside research with the research model, adapted and expanded from Buchner and Riehle [13]

The first row in the figure describes which domains are mainly affected by the four stages of the research model. The financial assessment of inner source requires comprehensive computational tools and techniques (first domain). These resources are then utilized to facilitate economic assessment (second domain), which subsequently enables the integration of insights into actual business processes (third domain). Ultimately, the goal is to enhance overall inner source adoption (fourth domain).

The second row shows the research model as presented earlier (Buchner and Riehle [13]).

The third row of of the figure maps the ongoing, unpublished, or future research to the corresponding segments of the overall research model, representing various research topics that were and are explored. Indicated in bold are the aspects of the algorithm discussed in Section 5.3. In same stage as the cost calculation are potential profit allocation calculations which also can based on the work time estimations, as previously discussed in Connection E (Figure 6.1).

In addition to the previously presented work, several ongoing research aligns with the research model.

Framework for work time based economic assessment Still unpublished at the time of writing is an extension of the commit-based work time estimation algorithm. The so-called *framework for work time-based economic assessment* (Connection F in Figure 6.1) extends the algorithm presented in Section 5.3 to a flexible framework. The new framework includes not only commits but all types of data sources. It enables the processing of messages (e.g., mails, real-time chat), calendar data, and issue-data for various use cases.

The term "framework" indicates that the goal is not only to estimate work time spent on development. It provides guidelines for connecting various use cases based on the measured data. It enables the financial assessment of processes and workflows connected to inner source development based on the estimated efforts. The framework also includes the organizational perspective, which was insufficient in the previous commit-based work time estimation algo-

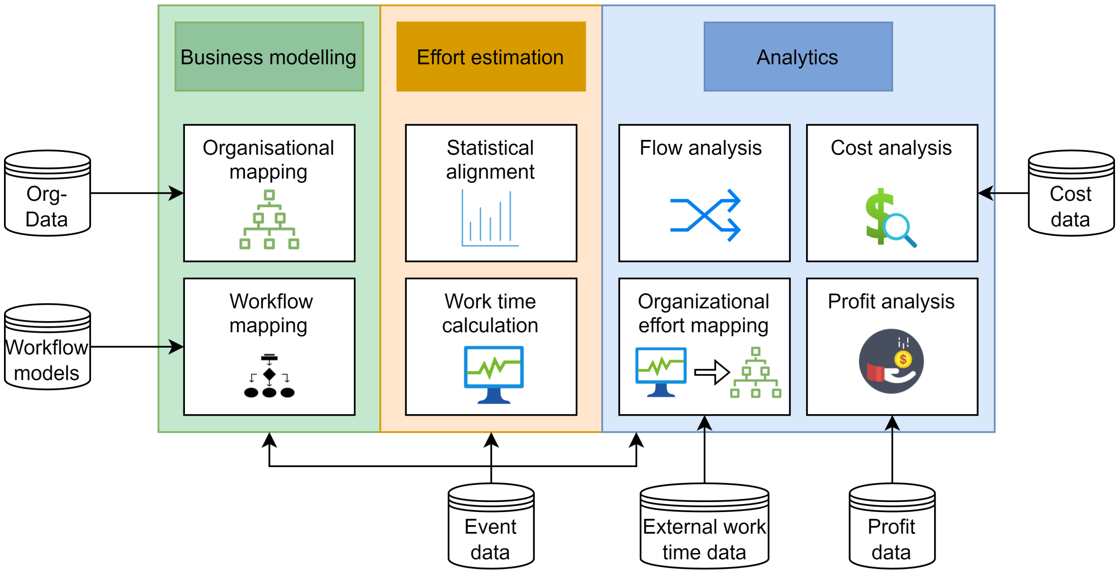

Figure 6.3: Overview of the Framework for Work Time Based Economic Assessment

rithm. By including detailed organizational data, it is easier to conduct communication analysis and identify collaboration patterns that emerge when organizations apply inner source, but developers and managers haven't actively recognized for themselves.

The framework unifies different business perspectives into one data-driven decision support model based on work time estimations. Implementing the framework into a tool in future research provides developers and managers with a practical multi-purpose tool for planning and conducting inner source.

Figure 6.3 shows the basic layout of the framework and the different aspects that it handles. The center of the new framework is an improved and adapted version of the work time estimation algorithm. On the left, the novel business modeling integration is visualized. The framework integrates workflows and organizational data. In combination with the effort estimation results (middle column of the figure), various analyses can be conducted (right side of the figure). This includes the already established cost calculation, but also profit allocation and calculations, IP flow analysis in inner source, and mapping the efforts to the organizational and workflow data.

From a research point of view, the framework crosses several stages of the research model (visualized in Figure 6.2). It sets computational basics with the adapted algorithm on which the economic measurement use cases (e.g. cost and profit calculation) are build. In the end, this also includes business process perspectives.

This framework shows that the work presented in the thesis sets important basics for a wide variety of inner source assessments. It helps answer the overall research question by providing an extended guideline and algorithm for conducting inner source measurements.

Modular Reimbursement Model: Additional ongoing research is focused on the development of a model capable of handling the calculation of tax contributions (transfer prices) among distinct business entities involved in software development. The algorithm introduced in Section 5.3 was specifically designed and evaluated for cost calculation purposes, utilizing the *cost plus* transfer pricing method by the OECD [47]. However, current transfer pricing models are in many cases inadequate for application in the inner source context.

We created a modular reimbursement model designed to compute payments (referred to as reimbursements) among various contributing and utilizing departments within software projects. This model is characterized by its adaptability and modularity, allowing it to be adapted to diverse organizational structures and costing principles commonly found within companies.

The modular reimbursement model builds upon the insights derived from the work time estimation algorithm we introduced (Connection G in Figure 6.1). It not only offers a framework for accountants and tax experts to compute the tax implications of inner source contributions but also takes into consideration the potential value of code use and reuse, in contrast to the cost-based approach outlined in this thesis.

This model integrates into the *business process embedding* phase of the research model (Figure 6.2) and lays the foundation for a more transparent evaluation of IP-contributions in day-to-day business operations.

Productivity Measurement: In a separate, yet unpublished, research related to inner source measurement, we are working on productivity measurement (Connection H in Figure 6.1). Our overarching objective is to measure the productivity of inner source in comparison to traditional software development. This is achieved by initially compiling an analysis of software engineering productivity indicators. Subsequently, we establish a productivity measurement methodology, building upon the details presented in Thomas Wolter's master's thesis [63], which illustrates the specific productivity metrics involved.

The productivity measurements we are currently developing fits into the *inner source adoption* stage of the research model (see Figure 6.2).

Managerial Decision Models: Measuring inner source presents an opportunity to enhance management accounting techniques. Hirsch and Riehle [31] proposed two conceptual models for management accounting within inner source (Connection I in Figure 6.1). Specifically, they introduced a compensation model, which forms the foundation of the modular reimbursement model discussed earlier. Additionally, they presented a model designed to help managers in evaluating the viability of inner source projects.

These inner source decision models align with the *business process embedding* stage of the research model (Figure 6.2).

Unified data model: Inner source measurement encounters a significant technical challenge in the manner in which data analysis and assessments are performed. The work time estimation algorithm as discussed in this paper relies on commit data. While it utilizes a single data source (commits), other metrics and tools mentioned in this section rely on diverse data sources and types. Consequently, each metric, tool, or model is implemented using a different subset of data with varying formats.

The development of a unified data model holds the potential to merge diverse data sources and formats. This unified model facilitates the execution of various inner source metrics and measurements. We are currently in the process of constructing such a unified data model for inner source measurement. A unified data model also helps in designing a suite of measurement tools that can enhance a company's processes and its software development through inner source.

As shown, the research model spans a broad spectrum of ongoing, recent, and unpublished research in the inner source measurement domain. Each research aspect presented here plays a role in the overall inner source research process, contributing to answering the overarching research question of this dissertation. The thesis aims to address how to quantify inner source for use within business processes. The literature analysis establishes the theoretical groundwork by examining the suitability of specific algorithmic approaches for the inner source domain. The research model outlines the progression of research when conducting inner source measurement-related studies. The work time estimation algorithm presented serves as an initial example of the research model. Currently ongoing research expands on it, demonstrates the viability and usability of the proposed research model.

6.6 Future Research

Based on the research model and other previous research, several topics of interest for future exploration can be identified.

One example is the development of predictive models for inner source (Hypothesis 2 of the research model). In this context, the utilization of a machine learning model for business-internal processes appears suitable, as indicated by our thematic analysis. Additionally, future research should create additional measurement algorithms and models that complement the existing work time estimation algorithm and the forthcoming framework.

From a business process perspective, further research should look deeper into strategic management decisions influenced by inner source. This may involve enhancing existing or creating new decision models. Future research should also develop more tools and models to support the managerial processes.

In addition to strategic processes, there is a need to enhance operational tasks when implementing inner source. To be more specific, future research should develop comprehensive transfer pricing models in the context of inner source, extending beyond the cost-based approach and modular reimbursement model discussed here.

Moreover, future research should conduct inner source adoption studies, not limited to the algorithm presented in this thesis, but also incorporating ongoing research, particularly the productivity metrics being developed. This approach will help determining the conditions under which inner source adoption is higher (e.g., organizational principles, applied development guidelines, established management processes). Measuring inner source productivity will also help in identifying the actual impact of inner source on a company.

6.7 Key Findings

We are able to answer the overall research question of this dissertation by summarizing the individual results we presented so far. The general goals of this thesis as we defined them in Section 3 were achieved by addressing four distinct sub-research questions (P1-RQ, P2-RQ1, P2-RQ2, and P3-RQ) in the three articles, followed by a comprehensive discussions.

From a business perspective, our findings indicate that inner source significantly influences a wide range of business processes, including development-related aspects, accounting, and managerial processes. Notably, many of these processes are currently not aligned with the high frequency at which inner source contributions cross organizational boundaries. Through our literature review, we gained a comprehensive understanding of how inner source impacts businesses. Such an understanding is vital for the development of tools and measures that can maximize the benefits of inner source.

Moreover, our research revealed that various tools and techniques can be employed to quantify inner source. Through thematic analysis, we identified algorithmic procedures that are particularly well-suited for application within the inner source domain. Approaches that allow for the assessment of individual IP contributions (e.g. commits or system interactions) emerged as the most suitable for inner source quantification. The majority of the identified tools and techniques are suitable as supportive approaches for inner source assessment. Certain methodologies, such as the classification of part- or full-time developers [54], can effectively complement the presented work time algorithm. This classification helps to identify the most viable algorithms and lays the groundwork for a large variety of inner source assessment tools yet to be developed.

To answer the research question it is important to establish a connection between the theoretical foundations of the SLR and practical applicability. Thus, we designed an research model, which shows a path how to progress from a research perspective in inner source measurement and quantifying its impact on business processes. This research model illustrates that both historical and predictive algorithms and methods must be developed, forming the basis for strategic and operational tools in the realm of inner source. The research model helps in improving the adoption of inner source in the long run.

In terms of technical implementation, any inner source quantification should incorporate the suitability analysis conducted during the SLR. We brought the theoretical groundwork of the SLR and research model into practice by creating an algorithm capable of assessing IP transfer (commits). The algorithm we introduced demonstrates the feasibility of measuring and financially evaluating inner source contributions. While our solution was primarily evaluated based on cost-based calculations, it lays the groundwork for the ongoing research initiatives we presented. The majority of the ongoing research utilizes the work time estimation algorithm or complement it to financially assess inner source development.

In our work, we selected the calculation of cost-based transfer prices as an illustrative implementation example. Through tax calculations, we demonstrated the practical applicability of the algorithm. This use case clarifies the integral role of the algorithm in addressing the research question. This also sets the basics for a wide variety of cost based metrics and processes in the inner source domain.

7

Conclusion and Outlook

In this thesis, we enable the quantification of inner source for its application in business processes, presenting several significant contributions toward this goal.

From a research perspective, this thesis establishes the theoretical groundwork for a comprehensive measurement of inner source. We identified the impact of inner source on business processes and explored tools and techniques for its quantification. The analysis of these tools and techniques revealed that while some basic algorithmic principles for inner source measurement exist, a comprehensive solution is yet to be developed.

Furthermore, we developed an inner source research model, designed to direct current and future research efforts towards a unified understanding in the field of inner source measurement. Additionally, we introduced an algorithm that estimates the time spent on software development, facilitating the quantification of code contributions for use in cost calculations.

Beyond the algorithm, we highlighted the interdependence of other ongoing, previously published, and unpublished research that builds upon the presented research model and work time estimation algorithm. For instance, we discussed the extension of the algorithm to a framework for more general work time-based inner source quantification. This framework has the capability to handle not only commits but also a wide range of software development-related and organizational data. The algorithm, along with the ongoing published and unpublished research, emphasizes the diverse research topics currently being explored. These expand the current knowledge outlined in this thesis to various domains, ranging from transfer pricing to business internal reimbursements for code contributions and productivity measurements in inner source.

Overall, this dissertation not only establishes essential theoretical and practical foundations but also paves the way for a broad spectrum of future research concerning the measurement and quantification of inner source. The ultimate objective of this research is to promote the adoption of inner source within companies. The findings presented in this thesis aid businesses in adapting to the open collaborative approach to software development, thereby fostering more efficient software development, cost reduction, and increased developer satisfaction - key benefits often associated with inner source initiatives [15].

References

[1] InnerSource Commons. 2021. State of InnerSource 2021. *https://innersourcecommons.org/learn/research/state-of-innersource-survey-2021/* Retrieved: 2023-07-12.

[2] InnerSource Commons. 2023. Community. *https://innersourcecommons.org/community/* Retrieved: 2023-03-21.

[3] InnerSource Commons. 2023. State of InnerSource 2023. *https://innersourcecommons.org/learn/research/state-of-innersource-survey-2022-2023/* Retrieved: 2023-10-05.

[4] Ž Antolić. 2008. An example of using key performance indicators for software development process efficiency evaluation. *R&D Center Ericsson Nikola Tesla* 6 (2008), 1–6. *https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=31f0cd36fa96ed46e7f5258e85aa2314d7d1545e*

[5] Saulius Astromskis, Andrea Janes, Alberto Sillitti, and Giancarlo Succi. 2014. An Approach to Non-invasive Cost Accounting. In *2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications*. IEEE, Verona, Italy, 30–37. *https://doi.org/10.1109/SEAA.2014.53*

[6] Bunea Bontas Cristina Aurora. 2013. The Cost Of Production Under Direct Costing And Absorption Costing – A Comparative Approach. *Annals - Economy Series* 2 (2013), 123–129. *https://www.utgjiu.ro/revista/ec/pdf/2013-02/19_Bunea%20Bontas%20Cristina.pdf*

[7] Victor Basili, Mikael Lindvall, Myrna Regardie, Carolyn Seaman, Jens Heidrich, Jürgen Münch, Dieter Rombach, and Adam Trendowicz. 2010. Linking Software Development and Business Strategy Through Measurement. *Computer* 43, 4 (05 2010), 57–65. *https://doi.org/10.1109/MC.2010.108*

[8] Saurabh Bilgaiyan, Santwana Sagnika, Samaresh Mishra, and M.N Das. 2017. A Systematic Review on Software Cost Estimation in Agile Software Development. *JOURNAL OF ENGINEERING SCIENCE AND TECHNOLOGY REVIEW* 10, 4 (08 2017), 51–64. *https://doi.org/10.25103/jestr.104.08*

[9] Barry W. Boehm. 1984. Software Engineering Economics. *IEEE Transactions on Software Engineering* SE-10, 1 (1984), 4–21. *https://doi.org/10.1109/TSE.1984.5010193*

[10] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (01 2006), 77–101. *https://doi.org/10.1191/1478088706qp063oa*

[11] Stefan Buchner. 2021. *A real-world example of Transfer Pricing in Inner Source*. Master's thesis. Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany. *https://oss.cs.fau.de/2021/05/27/final-thesis-a-real-world-example-of-transfer-pricing-in-inner-source/*

[12] Stefan Buchner and Dirk Riehle. 2022. Calculating the Costs of Inner Source Collaboration by Computing the Time Worked. In *Proceedings of the 55th Hawaii International Conference on System Sciences, HICSS '22*. ScholarSpace, virtual, 7466–7475. *https://doi.org/10.24251/HICSS.2022.896*

[13] Stefan Buchner and Dirk Riehle. 2023. A Research Model for the Economic Assessment of Inner Source Software Development. In *Proceedings of the 56th Hawaii International Conference on System Sciences, HICSS '23*. ScholarSpace, Maui, Hawaii, 353–364. *https://hdl.handle.net/10125/102672*

[14] Stefan Buchner and Dirk Riehle. 2023. The Business Impact of Inner Source and How to Quantify It. *ACM Comput. Surv.* 56, 2, Article 47 (sep 2023), 27 pages. *https://doi.org/10.1145/3611648*

[15] Maximilian Capraro. 2020. *Measuring Inner Source Collaboration*. Ph.D. Dissertation. Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU). *https://opus4.kobv.de/op*

us4-fau/frontdoor/index/index/start/2/rows/20/sortfield/score/sortorder/desc/searchtype/simple/query/capraro/docId/14005

[16] Maximilian Capraro, Michael Dorner, and Dirk Riehle. 2018. The Patch-Flow Method for Measuring Inner Source Collaboration. In *Proceedings of the 15th International Conference on Mining Software Repositories* (Gothenburg, Sweden) *(MSR '18)*. Association for Computing Machinery, New York, NY, USA, 515–525. *https://doi.org/10.1145/3196398.3196417*

[17] Maximilian Capraro and Dirk Riehle. 2016. Inner Source Definition, Benefits, and Challenges. *ACM Comput. Surv.* 49, 4, Article 67 (Dec. 2016), 36 pages. *https://doi.org/10.1145/2856821*

[18] Noel Carroll, Lorraine Morgan, and Kieran Conboy. 2018. Examining the Impact of Adopting Inner Source Software Practices. In *Proceedings of the 14th International Symposium on Open Collaboration* (Paris, France) *(OpenSym '18)*. Association for Computing Machinery, New York, NY, USA, Article 6, 7 pages. *https://doi.org/10.1145/3233391.3233530*

[19] Tjan-Hien Cheng, Slinger Jansen, and Marc Remmers. 2009. Controlling and monitoring agile software development in three dutch product software companies. In *2009 ICSE Workshop on Software Development Governance*. IEEE, Vancouver, Canada, 29–35. *https://doi.org/10.1109/SDG.2009.5071334*

[20] Melvin E. Conway. 1968. How Do Committees Invent? *Datamation* 14, 4 (April 1968), 28 – 31. *https://www.melconway.com/Home/pdf/committees.pdf*

[21] Tapajit Dey, Willem Jiang, and Brian Fitzgerald. 2022. Knights and Gold Stars: A Tale of InnerSource Incentivization. *IEEE Software* 39, 6 (11 2022), 88–98. *https://doi.org/10.1109/MS.2022.3192647*

[22] Jamie Dinkelacker, Pankaj K. Garg, Rob Miller, and Dean Nelson. 2002. Progressive Open Source. In *Proceedings of the 24th International Conference on Software Engineer-*

ing (Orlando, Florida) *(ICSE '02)*. Association for Computing Machinery, New York, NY, USA, 177–184. *https://doi.org/10.1145/581339.581363*

[23] Santiago Dueñas, Valerio Cosentino, Jesus M. Gonzalez-Barahona, Alvaro del Castillo San Felix, Daniel Izquierdo-Cortazar, Luis Cañas-Díaz, and Alberto Pérez García-Plaza. 2021. GrimoireLab: A toolset for software development analytics. *PeerJ Computer Science* 7 (July 2021), e601. *https://doi.org/10.7717/peerj-cs.601*

[24] Christof Ebert. 2014. Software Product Management. *IEEE Software* 31, 3 (2014), 21–24. *https://doi.org/10.1109/MS.2014.72*

[25] Christof Ebert, Bvs Krishna Murthy, and Namo Narayan Jha. 2008. Managing Risks in Global Software Engineering: Principles and Practices. In *2008 IEEE International Conference on Global Software Engineering*. IEEE, Bangalore, India, 131–140. *https://doi.org/10.1109/ICGSE.2008.12*

[26] Remo Eckert, Sathya Kay Meyer, and Matthias Stuermer. 2017. How Are Open Source Practices Possible within a Medical Diagnostics Company? Developing and Testing a Maturity Model of Inner Source Implementation. In *Proceedings of the 13th International Symposium on Open Collaboration* (Galway, Ireland) *(OpenSym '17)*. Association for Computing Machinery, New York, NY, USA, Article 3, 8 pages. *https://doi.org/10.1145/3125433.3125447*

[27] Henry Edison, Noel Carroll, Lorraine Morgan, and Kieran Conboy. 2020. Inner Source Software Development: Current Thinking and an Agenda for Future Research. *Journal of Systems and Software* 163 (05 2020), 110520. *https://doi.org/10.1016/j.jss.2020.110520*

[28] Robert C. Ford and W. Alan Randolph. 1992. Cross-Functional Structures: A Review and Integration of Matrix Organization and Project Management. *Journal of Management* 18, 2 (1992), 267–294. *https://doi.org/10.1177/014920639201800204* arXiv:https://doi.org/10.1177/014920639201800204

[29] R. Fuller. 2019. Functional Organization of Software Groups Considered Harmful. In *2019 IEEE/ACM International Conference on Software and System Processes (ICSSP)*. IEEE, Montreal, Canada, 120–124. *https://doi.org/10.1109/ICSSP.2019.00024*

[30] Georgios Gousios, Eirini Kalliamvakou, and Diomidis Spinellis. 2008. Measuring Developer Contribution from Software Repository Data. In *Proceedings of the 2008 International Working Conference on Mining Software Repositories* (Leipzig, Germany) *(MSR '08)*. Association for Computing Machinery, New York, NY, USA, 129–132. *https://doi.org/10.1145/1370750.1370781*

[31] Julian Hirsch and Dirk Riehle. 2022. Management Accounting Concepts for Inner Source Software Engineering. In *Lecture Notes in Business Information Processing* (2022-11-08/2022-11-11), Noel Carroll, Anh Nguyen-Duc, Xiaofeng Wang, and Viktoria Stray (Eds.), Vol. 463 LNBIP. Springer Science and Business Media Deutschland GmbH, Bolzano, Italy, 101–116. *https://doi.org/10.1007/978-3-031-20706-8_7*

[32] Martin Ivarsson and Tony Gorschek. 2011. A method for evaluating rigor and industrial relevance of technology evaluations. *Empirical Software Engineering* 16, 3 (June 2011), 365–395. *https://doi.org/10.1007/s10664-010-9146-4*

[33] Capers Jones. 2004. Software project management practices: Failure versus success. *CrossTalk: The Journal of Defense Software Engineering* 17, 10 (2004), 5–9. *https://www.inf.ufsc.br/~joao.dovicchi/pos-ed/pos/gerti/artigos/0410Jones.pdf*

[34] Sungjoo Kang, Okjoo Choi, and Jongmoon Baik. 2010. Model-Based Dynamic Cost Estimation and Tracking Method for Agile Software Development. In *2010 IEEE/ACIS 9th International Conference on Computer and Information Science*. IEEE, Yamagata, Japan, 743–748. *https://doi.org/10.1109/ICIS.2010.126*

[35] Hrvoje Karna. 2020. Data Mining Approach to Effort Modeling On Agile Software Projects. *Informatica* 44, 2 (06 2020), 231–239. *https://doi.org/10.31449/inf.v44i2.2759*

[36] Barbara Kitchenham. 2004. Procedures for performing systematic reviews. *Keele, UK, Keele University* 33, 2004 (2004), 1–26. *https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=29890a936639862f45cb9a987dd599dce9759bf5*

[37] Martin Kornberger, Dane Pflueger, and Jan Mouritsen. 2017. Evaluative infrastructures: Accounting for platform organization. *Accounting, Organizations and Society* 60 (2017), 79–95. *https://doi.org/10.1016/j.aos.2017.05.002*

[38] Y.H. Kwak and J. Stoddard. 2004. Project risk management: lessons learned from software development environment. *Technovation* 24, 11 (2004), 915–920. *https://doi.org/10.1016/S0166-4972(03)00033-6*

[39] Juho Lindman, Matti Rossi, and Pentti Marttiin. 2008. Applying Open Source Development Practices Inside a Company. In *Open Source Development, Communities and Quality*, Barbara Russo, Ernesto Damiani, Scott Hissam, Björn Lundell, and Giancarlo Succi (Eds.). Springer US, Boston, MA, 381–387. *https://doi.org/10.1007/978-0-387-09684-1_36*

[40] Guy Martin and Aaron Lippold. 2011. Forge.mil: A Case Study for Utilizing Open Source Methodologies Inside of Government. In *Open Source Systems: Grounding Research*, Scott A. Hissam, Barbara Russo, Manoel G. de Mendonça Neto, and Fabio Kon (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 334–337. *https://doi.org/10.1007/978-3-642-24418-6_28*

[41] Orly Mazur. 2016. Transfer Pricing Challenges in the Cloud. *Boston College Law Review* 57, 2 (2016), 643–693. *https://lawdigitalcommons.bc.edu/bclr/vol57/iss2/6*

[42] William E. McCarthy. 1982. The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment. *The Accounting Review* 57, 3 (1982), 554–578. *http://www.jstor.org/stable/246878*

[43] Lorraine Morgan, Joseph Feller, and Patrick Finnegan. 2011. Exploring inner source as a form of intraorganisational open innovation. In *19th European Conference on Infor-*

mation Systems, ECIS, Virpi Kristiina Tuunainen, Matti Rossi, and Joe Nandhakumar (Eds.). AISeL, Helsinki, Finland, 151. *http://aisel.aisnet.org/ecis2011/151*

[44] Donatien Koulla Moulla, Alain Abran, and Kolyang. 2021. Duration Estimation Models for Open Source Software Projects. *International Journal of Information Technology and Computer Science* 13, 1 (02 2021), 1–17. *https://doi.org/10.5815/ijitcs.2021.01.01*

[45] A. Neus and P. Scherf. 2005. Opening minds: Cultural change with the introduction of open-source collaboration methods. *IBM Systems Journal* 44, 2 (2005), 215–225. *https://doi.org/10.1147/sj.442.0215*

[46] OECD. 2015. *Aligning Transfer Pricing Outcomes with Value Creation, Actions 8-10 - 2015 Final Reports*. OECD, Paris, France. 192 pages. *https://doi.org/10.1787/9789264241244-en*

[47] OECD. 2022. *OECD Transfer Pricing Guidelines for Multinational Enterprises and Tax Administrations 2022*. OECD, Paris, France. 659 pages. *https://doi.org/10.1787/0e655865-en*

[48] Marcel Olbert and Christoph Spengel. 2017. International taxation in the digital economy : challenge accepted? *World Tax Journal : WTJ* 9, 1 (2017), 3–46. *https://madoc.bib.uni-mannheim.de/41867/*

[49] Ken Peffers, Tuure Tuunanen, Marcus Rothenberger, and S. Chatterjee. 2007. A design science research methodology for information systems research. *Journal of Management Information Systems* 24 (01 2007), 45–77.

[50] Christian Plesner Rossing, Martine Cools, and Carsten Rohde. 2017. International transfer pricing in multinational enterprises. *Journal of Accounting Education* 39, C (2017), 55–67. *https://EconPapers.repec.org/RePEc:eee:joaced:v:39:y:2017:i:c:p:55-67*

[51] Fumin Qi, Xiao-Yuan Jing, Xiaoke Zhu, Xiaoyuan Xie, Baowen Xu, and Shi Ying. 2017. Software effort estimation based on open source projects: Case study of Github.

Information and Software Technology 92 (2017), 145–157. *https://doi.org/10.1016/j.infsof.2017.07.015*

[52] Dirk Riehle, Maximilian Capraro, Detlef Kips, and Lars Horn. 2016. Inner Source in Platform-Based Product Engineering. *IEEE Transactions on Software Engineering* 42, 12 (12 2016), 1162–1177. *https://doi.org/10.1109/TSE.2016.2554553*

[53] Dirk Riehle, John Ellenberger, Tamir Menahem, Boris Mikhailovski, Yuri Natchetoi, Barak Naveh, and Thomas Odenwald. 2009. Open Collaboration within Corporations Using Software Forges. *IEEE Software* 26, 2 (2009), 52–58. *https://doi.org/10.1109/MS.2009.44*

[54] Gregorio Robles, Jesús M. González-Barahona, Carlos Cervigón, Andrea Capiluppi, and Daniel Izquierdo-Cortázar. 2014. Estimating Development Effort in Free/Open Source Software Projects by Mining Software Repositories: A Case Study of OpenStack. In *Proceedings of the 11th Working Conference on Mining Software Repositories* (Hyderabad, India) *(MSR 2014)*. Association for Computing Machinery, New York, NY, USA, 222–231. *https://doi.org/10.1145/2597073.2597107*

[55] Geoffrey G Roy. 2004. A risk management framework for software engineering practice. In *2004 Australian Software Engineering Conference. Proceedings.* IEEE, Melbourne, Victoria, Australia, 60–67. *https://doi.org/10.1109/ASWEC.2004.1290458*

[56] Klaas-Jan Stol, Muhammad Ali Babar, Paris Avgeriou, and Brian Fitzgerald. 2011. A comparative study of challenges in integrating Open Source Software and Inner Source Software. *Information and Software Technology* 53, 12 (2011), 1319–1336. *https://doi.org/10.1016/j.infsof.2011.06.007*

[57] Klaas-Jan Stol and Brian Fitzgerald. 2015. Inner Source–Adopting Open Source Development Practices in Organizations: A Tutorial. *IEEE Software* 32, 4 (2015), 60–67. *https://doi.org/10.1109/MS.2014.77*

[58] The International Federation of Accountants. 2009. *Evaluating and Improving Costing in Organizations*. International Federation of Accountants (IFAC), New York,

USA. *https://www.ifac.org/knowledge-gateway/preparing-future-ready-professionals/publications/evaluating-and-improving-costing-organizations*

[59] Richard Torkar, Pau Minoves, and Janina Garrigós. 2011. Adopting free/libre/open source software practices, techniques and methods for industrial use. *Journal of the Association for Information Systems* 12, 1 (2011), 89–122. *https://doi.org/10.17705/1jais.00252*

[60] United Nations. 2014. *United Nations Practical Manual on Transfer Pricing for Developing Countries*. United Nations, New York, USA. *https://www.un-ilibrary.org/content/books/9789210561372*

[61] Muhammad Usman, Emilia Mendes, Francila Weidt, and Ricardo Britto. 2014. Effort estimation in agile software development. In *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*. ACM, Turin, Italy, 82–91. *https://doi.org/10.1145/2639490.2639503*

[62] J.M. Verner and N. Cerpa. 2005. Australian software development: what software project management practices lead to success?. In *2005 Australian Software Engineering Conference*. IEEE, Brisbane, Australia, 70–77. *https://doi.org/10.1109/ASWEC.2005.14*

[63] Thomas Wolter. 2022. *Creating, Measuring and Evaluating a Metric for Software Engineering Performance*. Master's thesis. Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany. *https://oss.cs.fau.de/2023/02/10/final-thesis-creating-measuring-and-evaluating-a-metric-for-software-engineering-performance/*

[64] Hong Wu, Lin Shi, Celia Chen, Qing Wang, and Barry Boehm. 2016. Maintenance Effort Estimation for Open Source Software: A Systematic Literature Review. In *2016 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, Raleigh, NC, USA, 32–43. *https://doi.org/10.1109/ICSME.2016.87*

A

The Business Impact of Inner Source and How to Quantify It

In the following, the full text of the article *The Business Impact of Inner Source and How to Quantify It* can be found. The article was originally published as:

[14] Stefan Buchner and Dirk Riehle. 2023. The Business Impact of Inner Source and How to Quantify It. ACM Comput. Surv. 56, 2, Article 47 (sep 2023), 27 pages. https://doi.org/10.1145/3611648

The Business Impact of Inner Source and How to Quantify It

STEFAN BUCHNER and DIRK RIEHLE, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

Inner source software development is the practice of using open source practices for firm-internal software development. Practitioner reports have shown that inner source can increase flexibility and reduce costs. Despite the potential benefits of inner source, there has been little research on its impact on businesses and their processes. To address this gap, we conducted a systematic literature review that identified which business processes are affected by inner source development, particularly within the accounting and management domain. Our review revealed the need for new dedicated community building processes within companies. In addition, we examined computational tools and techniques that can be used to measure inner source development. We found that existing tools and techniques are insufficiently suitable to manage inner source processes. Based on this, we propose research topics for future work on quantifying inner source.

CCS Concepts: • **General and reference** → **Surveys and overviews**; *Measurement;*; *Metrics* • **Software and its engineering** → Reusability; **Collaboration in software development;** • **Applied computing** → *Cross-organizational business processes*; **Business-IT alignment;** • **Social and professional topics** → *Project and people management*; *Transborder data flow*;

Additional Key Words and Phrases: Inner source, open source, internal open source, software engineering, software development, business processes, cost estimation, effort estimation, cost calculation, accounting, taxation, transfer pricing

ACM Reference format:
Stefan Buchner and Dirk Riehle. 2023. The Business Impact of Inner Source and How to Quantify It. *ACM Comput. Surv.* 56, 2, Article 47 (September 2023), 27 pages.
https://doi.org/10.1145/3611648

1 INTRODUCTION

Reacting flexibly to changes in rapidly evolving markets is crucial for companies that develop software as a core part of their products. Although agile software development makes it easier to adapt to customer needs, it often lacks general organizational flexibility, prompting companies to seek more agility on a larger organizational scale by adopting open-source principles for firm-internal software development [19].

Using open-source principles for internal development work is called inner source [60]. Rather than involving developers from outside the company, organizations apply the methods used in

This work was supported by the DFGs (German Research Foundation) Research Grants Program (Management Accounting for Inner Source, RI 2147/5-2).
Authors' address: S. Buchner and D. Riehle, Friedrich-Alexander-Universität Erlangen-Nürnberg, Open Source Research Group, Martensstr. 3, Erlangen 91058, Germany; emails: stefan.buchner@fau.de, dirk@riehle.org.

0360-0300/2023/09-ART47 $15.00
https://doi.org/10.1145/3611648

open-source development to enable people to develop and improve projects or modules that are internally (as far as possible) unrestrictedly available [30]. Inner source development is closely incorporated into internal review cycles with early and frequent feedback, enabling close collaboration across organizational boundaries [23].

Applying inner source to a company's development process and organizational structure can provide numerous advantages, even if not all of them are directly measurable. Inner source can lead to higher-quality software components due to input and reviews from previously unincorporated teams, better knowledge sharing, and increased employee satisfaction [11, 55, 60]. Moreover, inner source can facilitate component reuse, a characteristic of open-source software [52], which can make platform development more efficient [55]. This, in turn, can lead to a more streamlined development process that reduces time-to-market and lowers costs, as several studies have demonstrated [13, 23].

The practical relevance of inner source is evident in the InnerSource Commons, an organization where both IT and non-IT companies come together to share and derive benefits from their experiences with inner source. According to a survey conducted by the InnerSource Commons, there is significant interest in inner source across a wide range of industry sectors. For instance, 37.8% of the respondents worked in the technology sector, while nearly 19.5% were from financial services, and 13.4% represented healthcare and pharmaceuticals sectors [1]. Due to its numerous developmental and organizational benefits, inner source is gaining popularity in both academia and industry [23]. However, it is not yet widespread, and there are various reasons for this within different business domains. For example, management's understanding of inner source or the developers attitude toward knowledge sharing, as identified by Edison et al. [23]. Recent research has primarily focused on the cultural and operational aspects of inner source. While examining the existing economic advantages, such as cost reduction [13], might be a useful motivator for the widespread adoption of inner source, only a few papers have examined its impact on business processes outside of engineering and attempted to measure and quantify it. Previous research suggests that the barriers to introducing inner source can be high, particularly because it is still unclear how exactly inner source creates strategic economic value for companies [23].

From a developer's perspective, inner source development may include contributing code across any organizational boundaries [12, 13]. This type of development occurs frequently and involves unpredictable flows of IP across internal organizational boundaries, making it difficult to measure, quantify and predict for economic purposes [10]. However, as this article will demonstrate, existing tools and techniques are not easily applicable to the cross-boundary collaboration pattern of inner source.

Being able to economically assess and quantify inner source development can serve as the foundation for a wide range of business applications. For instance, Capraro [11] developed a basic model for measuring code contributions in inner source. Buchner and Riehle [10] demonstrated that economic assessment of inner source is feasible, although they focused on a taxation use case and a more generally applicable approach is necessary.

Failing to consider inner source quantification could result in significant harm, including illegal profit-shifting, as demonstrated in previous research [10]. This is due to cross-boundary code-flows between different taxable jurisdictions. The Organisation for Economic Cooperation and Development (OECD) has recognized software development as a significant challenge for tax officials [49, 50], and this is particularly relevant for inner source. Comprehensive measurement of inner source's IP-flows and impact on the business is crucial from a taxation perspective.

The purpose of this article is to examine the impact of inner source collaboration on a company's operational and strategic processes and to suggest methods for measuring this impact to enhance effectiveness and innovation. Although our focus is on inner source development, we believe that our findings can be applied to improve a wide range of business processes that are related to or influenced by software development within organizations.

Moreover, this article aims to address the background and limitations of current economic assessment tools and techniques for businesses, and why they are inadequate for the inner source use case. The article examines both theoretical and practical aspects, discussing computational tools and techniques (such as algorithms, models, and methods) for measuring and estimating software-related efforts or costs, as well as exploring how inner source software development affects processes within the industry. The ultimate objective is to identify the connections between previously disconnected topics of theoretical effort estimation algorithms and business processes, particularly for their application within inner source. The end result is a comprehensive and integrated view of the topic of economic inner source assessment, encompassing multiple perspectives. Overall, this article presents the following contributions:

— A survey of the effects of cross-boundary collaboration on businesses and their processes
— A survey of existing computational tools and techniques for measuring economic impact of inner source development
— An analysis showing how new computational tools and techniques can help solve existing challenges with inner source development and why no such algorithm exist yet
— A presentation of potential future research topics connected to economic inner source assessment.

The remainder of the article is structured as followed: Section 2 shows related work to economic assessment of inner source and explains the need for a review in more detail. Section 3 will give an overview of the research question of this article as well as the methodology used to answer it. Section 4 will then present the results of the conducted systematic literature review (SLR), followed by an in-depth discussion in Section 5 on how themes identified during the literature review are connected as well as their implications to current research. Section 6 will lastly outline how this article can contribute to future research followed by a conclusion.

2 RELATED WORK

Our goal is to understand the measurement of inner source development, how it affects business processes, and how to quantify it. We recognize that the economic assessment of inner source lies at the intersection of business processes and computational tools and techniques. Therefore, we will examine previous research in these two domains to gain a better understanding of the topic.

2.1 Previous Inner Source and Business Process Research

Numerous papers have defined inner source (e.g., [14, 17, 45, 55, 58–60]), their benefits and challenges (e.g., [13, 55, 59, 60]) or took industry aspects directly into consideration (e.g., [27, 46, 55]). Edison et al. [23] published a literature review on inner source definitions, benefits, challenges, as well as research gaps. In addition to inner source specific research, many business processes and practices are also well defined within traditional economic research (outside of inner source).

For the accounting domain, handling cost calculations (e.g., full absorption costing [5]) or account models for platforms [38] have already been established. Basic algorithms and system designs for comprehensive accounting with computer systems were established as early as 1982/1996 [26, 44]. In the realm of taxation, the OECD has established fundamental approaches that, as previously mentioned, are not entirely appropriate for software development [49–51, 63]. While the OECD has outlined the fundamental principles of taxation, these principles are becoming increasingly problematic when applied to software development, including inner source.

Various methods exist to measure a company's success and aid in making strategic and operational decisions in management. For instance, risk management is commonly approached [22, 39, 57]. However, inner source development presents a challenge to traditional management approaches due to its cross-boundary collaboration pattern. Those were designed for the application within one business entity only (e.g., cost calculation). Consequently, measuring costs, staff development, and processes become more challenging in inner source. Furthermore, while software management practices are well-established, they face new challenges when applied to inner source development, as this article will show. As a SLR, this research will closely examine the existing literature, extending the work presented in this section.

2.2 Open Topics

The concepts, advantages, and obstacles of inner source are well-established. However, research on measuring inner source collaboration is insufficient. Capraro identified inner source collaboration patterns [11], while Buchner and Riehle developed an algorithm to measure work time [10]. Initial research on inner source measurement for management accounting has also been conducted [31].

Edison et al. [23] noted a lack of metrics for measuring improvements resulting from inner source initiatives. While they identified several areas where further research is needed (such as management, inner source adoption, and methodologies), their proposal for measuring inner source impact was brief and did not address how companies can tackle measurement-related challenges.

Understanding the needs and basics for various inner source assessment metrics or tools helps answer the unsolved questions Edison et al. also proposed.

3 METHODOLOGY

This section gives an overview of the research question as well as the used methods.

3.1 Research Question and Goals

As previously presented, there has been limited research on the economic assessment of inner source development. Motivated by this gap, the present article aims to address the following research question:

RQ: What is the economic impact of inner source on companies and how can it be quantified?

To address our research question, we investigate the impact of inner source on business processes. In doing so, we examine existing computational tools and techniques that measure software development. We classify these tools and techniques and analyze their suitability for application within the inner source domain.

3.2 Outline of the Paper

The article is structured around two main perspectives: business processes and computational tools and techniques. Each perspective is considered separately before bringing them together.

Figure 1 provides a detailed overview of the parts of the article, including the methodology applied for each section, the results obtained, and the primary question addressed. These parts also serve as a roadmap for the article and illustrate how they build upon one another. The first level in the figure (Part 1 and 2; Section 4) presents the findings of the SLR, which includes an explanation of the codes and themes generated during thematic analysis. The second level (Parts 3a, 3b, 3c; Section 5) delves deeper into the artifacts resulting from the thematic analysis, examines their relationship, and discusses their implications for inner source. These insights can be utilized in future research to develop more robust economic inner source measurement models.

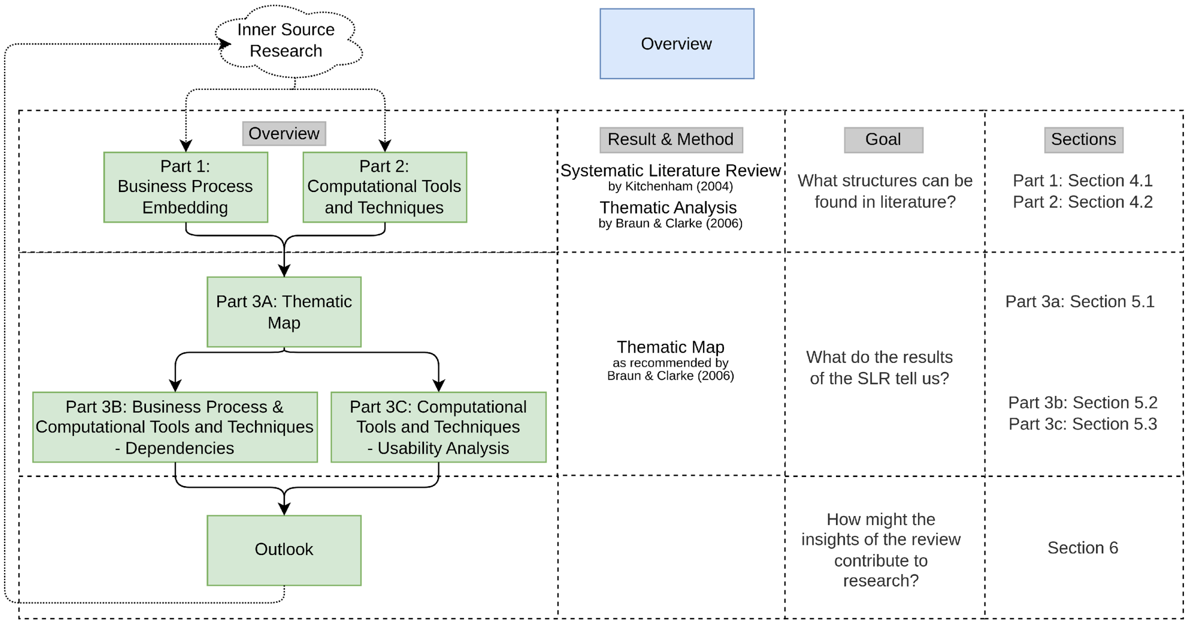

Fig. 1. Overview of the following sections, used methods, and results of the review.

In detail, the goals of the parts are as follows:

Part 1: In this section, we are analyzing existing literature to identify business processes and explore how they are impacted by inner source software development.

Part 2: Here, we focus on the computational tools and techniques that can be used to measure software development within businesses.

Part 3: We then demonstrate the interconnection between business processes, tools, and techniques by presenting:

- Part (3a) A high-level thematic map that illustrates the general relationship between the themes identified in our SLR.
- Part (3b) An in-depth excerpt of the high-level thematic map that explains how existing tools and techniques can be used to measure or support business processes.
- Part (3c) Another in-depth excerpt of the high-level thematic map that highlights the suitability of certain tools and techniques for application within the inner source domain.

3.3 Methodological Overview

3.3.1 Research Process. To address the research question, two primary research methods were employed: The SLR by Kitchenham [37] and thematic analysis by Braun and Clarke [9]. These methods were used in combination, as they complement each other. Kitchenham's SLR does not provide a detailed explanation of the data extraction and synthesis process, which is the main focus of Braun and Clarke's thematic analysis. Both methods emphasized a non-linear/iterative approach. We followed this approach by repeatedly searching and filtering research (as suggested by Kitchenham) and then analyzing the data using Braun and Clarke's thematic analysis as part of Kitchenham's process. Therefore, we adopted the basic research process proposed by Kitchenham.

Figure 2 shows in detail the performed research steps and how the two approaches work together. The figure is divided into three columns, with the first two columns outlining the SLR steps proposed by Kitchenham, while the third column illustrates the thematic analysis steps as per Braun and Clark.

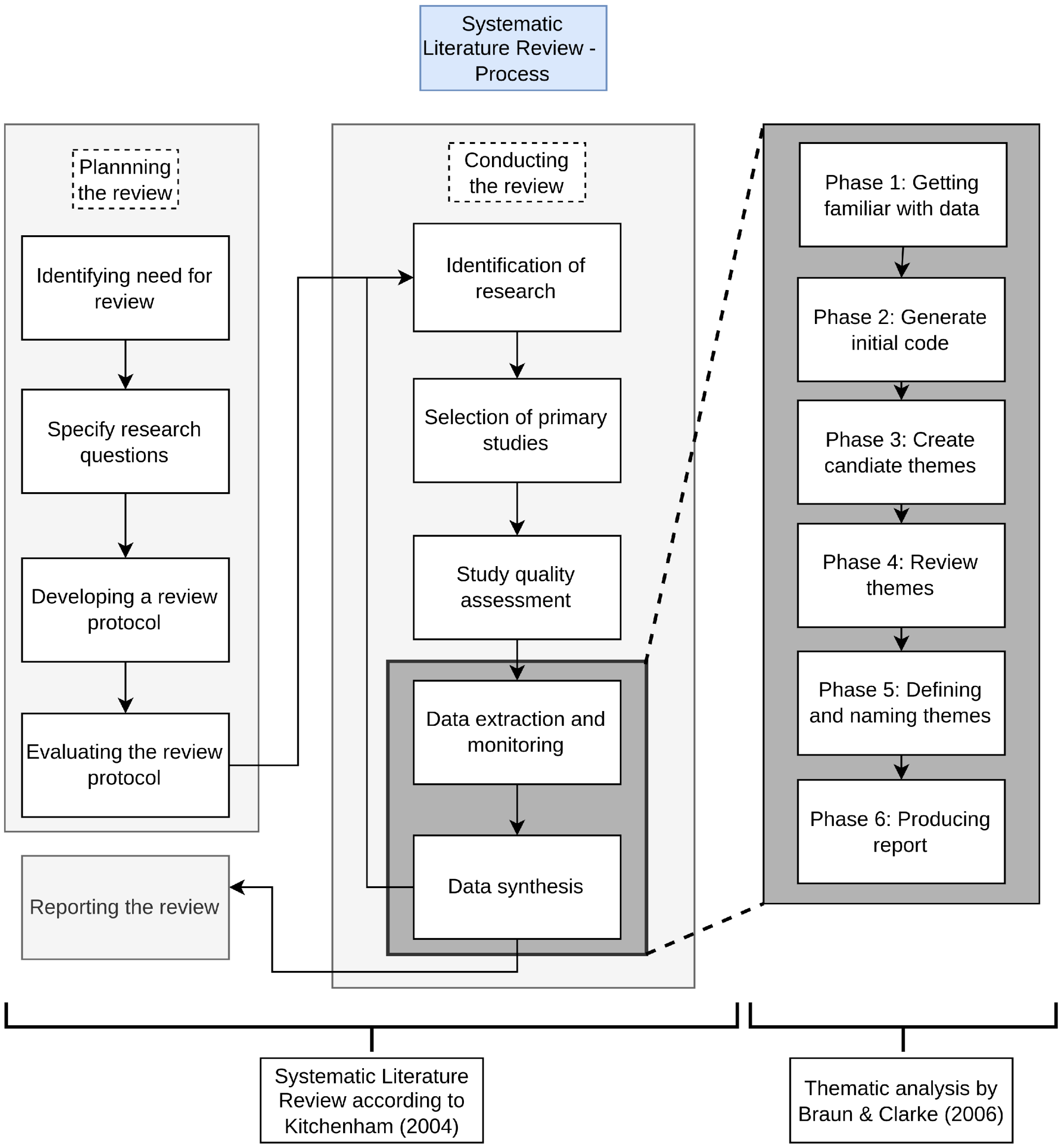

Fig. 2. Overview of the conducted research process based on Kitchenham [37] and Braun and Clarke [9].

The first column in our review outlines the planning steps, we undertook prior to conducting our review. These steps included analyzing the need for a review, specifying the research question(s), writing a review protocol, and evaluating it. In the following section, we provide more details about the review planning process, while the research question was already presented in a previous section.

The second column shows the iterative steps of the conducted literature review: Identifying the research, selecting studies, assessing their quality followed by the data extraction and synthesis step. Kitchenham specified the data extraction and synthesis descriptively and not in all-detail. We used the thematic analysis framework from Braun and Clarke for these two steps of Kitchenham to be able to conduct our research.

The third column then explains in detail the six thematic analysis steps for analyzing the literature data: Getting familiar with the data, generating an (initial) code system (characteristics identified within the literature), followed by creating, reviewing, and naming themes (logically grouped codes), which are in the last step reported. Based on the created themes, their dependencies, and missing aspects the next iterations were conducted.

We conducted three major iterations to address our research question. In our first iteration, we identified that economic inner source assessment is influenced by two perspectives: business processes and computational tools and techniques. Although they are not completely independent from each other, they originate from different research backgrounds. The economic perspective is located within business-process research, while the tools and techniques are primarily located within the computer science domain. Therefore, we searched for relevant research papers in both domains using different search terms and analyzed them with different goals in mind. The following two sections will provide more detail about the steps we performed.

3.3.2 Research Protocol. This section presents the details of the research protocol that we developed for the SLR. Our protocol follows the order proposed by Kitchenham [37], and includes all the necessary information about the various steps we conducted. We created the protocol during the planning phase of our literature review.

Need for review: We already explained the need for a review (first step in the process proposed by Kitchenham) in detail within the previous sections. The main motivation is that it is still unclear how inner source creates business value and the lack of metrics, therefore [11]. Previous research showed a low number of inner source tools and algorithmic procedures. Therefore, we conduct this review to identify how inner source impacts business processes and how existing tools and techniques can measure such impact.

Evaluation: As part of our SLR, we followed the evaluation process described by Kitchenham. First, each author independently created a review protocol including research questions, keywords, qualitative criteria, and inclusion/exclusion criteria. We then evaluated each other's protocols before discussing and creating a mutually agreed upon version. We used the same approach for the research process itself, with each author conducting independent searches and proposing key findings that were then discussed together. Additionally, we solicited outside reviews from researchers in our research group who were not extensively versed in the topic. By following this evaluation process, we ensured the rigor and thoroughness of our literature review.

Databases: We used the databases Google Scholar, IEEE Xplorer, ACM Digital Library, Springer Link, Ebscohost, Wiley, and Scopus.

Identification process: The initial iteration of the review involved searching for fundamental concepts related to economic assessment and business processes. An essential component of the literature identification process was the use of forward and backward searches, also known as snowballing, which aided in the discovery of additional relevant information for answering the research question.

During the first iteration, we recognized that our research spans across different fields of science, including economics and computer science, and has been published in various journals with different contexts, using a range of keywords and phrases. Therefore, we decided to split our search and iterations into two main topics: the business process side of inner source and the tools and techniques side, which includes algorithms, applications/tools, methods, and models.

During our research, we found papers concerning different levels: Some were more theoretical/algorithmic-based and aligned well with the inner source principle, some look rather at the broader business process impacts. Classifying those was part of the thematic analysis by Braun and Clarke, which helped to identify literature gaps for the next iteration.

We realized that few inner source papers exist, particularly in the economic domain, where most methods are discussed without considering the software development method. Hence, during our selection and quality assessment, we looked through papers that do not directly relate to inner source to understand the current research state of measuring cross-boundary collaboration. This influenced our choice of inclusion and exclusion criteria.

Keywords: Following the two perspectives of business processes and computational tools and techniques, we used different keywords to identify relevant literature. Throughout our iterative process, we were able to expand our search terms by adding additional keywords. For both domains of review, we identified a common set of keywords related to software development approaches:

(Inner source OR open-source OR collaborative development OR cross-boundary collaboration OR cross border collaboration OR internal open-source OR software engineering OR software development OR DevOps OR agile OR platform)

That general search term was combined with the search term for each of the review domains. For the business process domain, the search term was:

(Business processes OR management OR accounting OR controlling OR taxation OR transfer pricing OR organization OR businesses OR enterprises OR organizational principles OR organization forms OR absorption costing OR cost calculation OR project management OR risk management OR product management)

For the tools and techniques domain, we used the following search term:

(Software development OR programming OR ((cost OR effort) AND (calculation OR prediction OR estimation OR measuring OR quantifying OR computing OR calculating)) OR measurement OR KPI)

Quality criteria: The most crucial qualitative criteria for our study were the peer-review status of the papers and their publication in a recognized journal, conference, or as a well-defined (technical) report. For instance, algorithms must be comprehensible and reproducible, particularly in the case of reviewed tools and techniques. In addition to peer-review, we also considered papers originating from well-known organizations in the relevant domain, such as the OECD for transfer pricing [49, 50]. Moreover, we limited our scope to English-language papers. To assess papers, we used the rigor and relevance criteria proposed by Ivarsson and Gorschek [33] in their technology evaluation method: To assess the rigor of a article, we evaluated (as proposed) whether the overall research context, study design, and validity (as well as threats to it) were discussed and to which extent. We evaluated practical relevance by examining the context and determining the degree of industry relevance (e.g., no student projects).

Inclusion and exclusion criteria: We additionally examined all papers which fulfilled the qualitative criteria toward their usefulness for our particular review. We have done this by defining inclusion and exclusion criteria. We included papers which

— are connected to inner source measurement in general
— present tools or techniques for measuring or predicting business (process) related aspects
— address problems within businesses and their processes connected to cross-boundary collaboration
— calculate work effort or costs on different levels (code-level, project-level, or business-level)
— give noteworthy insights useful for measuring inner source and affected processes

We explicitly excluded papers which

— have no thematic connection or usability within inner source, cross-boundary collaboration in general, or relate business processes
— presented tools and techniques that are non-repeatable. That especially affects machine learning algorithms, mostly for cost calculation.
— presented tools and techniques which are not adaptable to inner source development (not able to assess cross-boundary IP-flow)

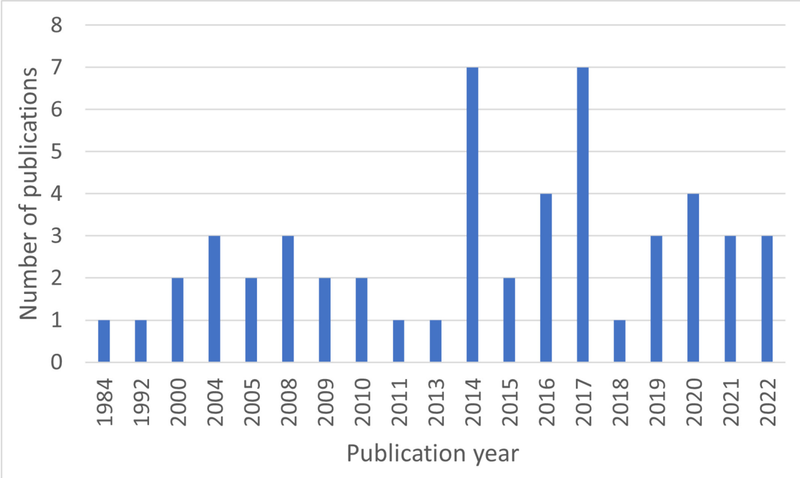

Fig. 3. Number of publications per year.

3.3.3 Thematic Analysis. We completed the data extraction and synthesis step of Kitchenham's SLR by using thematic analysis, as proposed by Braun and Clarke [9]. We followed the steps outlined in their methodology, as depicted in Figure 2. Our analysis encompassed two distinct perspectives: business processes and computational tools/techniques. To ensure comprehensive coverage, we identified important aspects, referred to as codes, and organized them into logical groupings, known as themes. This process was conducted independently for both domains.

Braun and Clarke [9] propose several approaches for thematic analysis. In this article, we chose a deductive approach [9], starting from the previously presented research question (economic inner source assessment) and analyze/extract step by step the information gathered by the previous iteration. In our case, we started to look into inner source assessment options and their business process influence first. Next, we analyzed the applicability to the inner source domain.

Braun and Clarke distinguish between semantic and latent approaches to thematic analysis. While semantic analysis focuses on the explicit meaning of written aspects, latent analysis attempts to uncover the meaning behind the written words [9]. For our study, it is important to cover both aspects, as general patterns and algorithms can be analyzed by examining the initial goals of the papers. However, to fully understand the economic implications of inner source measurements, we also need to look beyond the surface-level meaning of certain papers. We have chosen to use a combination of both approaches, as only a few papers (especially those related to business processes) were originally written with inner source in mind. These papers need to be analyzed more deeply using a latent approach to extract their core aspects that are applicable to inner source.

4 SYSTEMATIC LITERATURE REVIEW

The following two subsections present the results of the SLR, divided into two parts: Part 1 focuses on the business process view of inner source measurement, while Part 2 focuses on the computational tools and techniques view.

Our SLR yielded 52 relevant papers across the business and computational tools and techniques domains. Our analysis identified 7 themes and 27 codes within these papers. While a majority of the papers were published within the last decade, the topic of inner source has garnered significant attention in recent years. Nevertheless, we also included older papers as they laid the fundamental economic and measurement foundations that still hold true today. Figure 3 displays the number of publications per year.

Table 1. Overview of Sources Per Theme

Theme	# of Sources	Sources
A - Management processes	17	[3, 6, 13, 16, 21, 22, 31, 34, 39, 49, 55, 57, 58, 60, 62, 63, 65]
B - Accounting processes	21	[4, 5, 7, 10, 11, 15, 23, 30, 31, 36, 38, 43, 48–51, 53, 62–64, 67]
C - Development processes	17	[11, 13, 23–25, 28, 30–32, 40, 42, 49, 55, 58–60, 68]
D - Computation goals	12	[3, 4, 6–8, 16, 29, 36, 54, 56, 64, 67]
E - Algorithmic procedure	15	[3, 4, 6, 7, 10, 16, 20, 29, 35, 36, 47, 54, 56, 64, 67]
F - Data sources	19	[3, 4, 6, 10, 16, 20, 35, 36, 47]
G - Development context	3	[7, 8, 61]

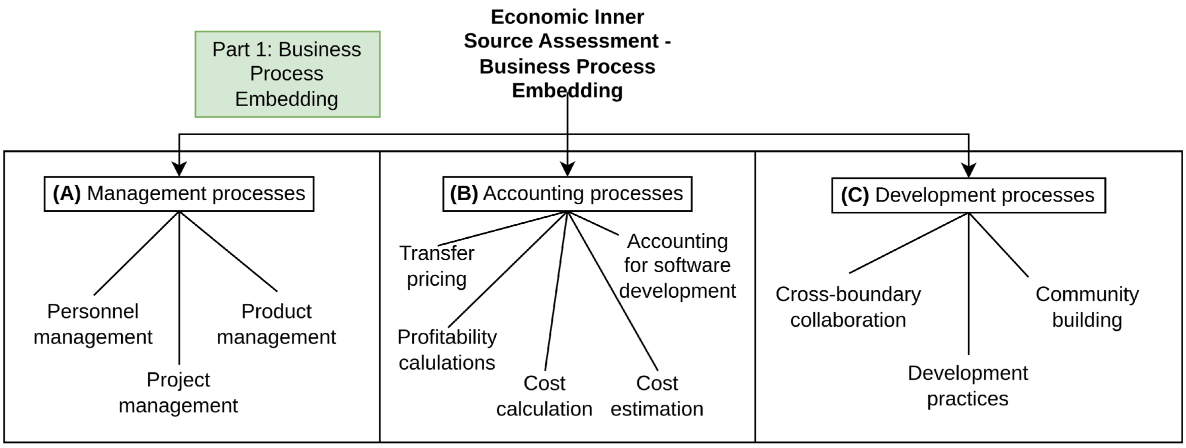

Fig. 4. Themes and codes on the business process perspective of measuring inner source.

Table 1 provides a concise overview of the sources used to identify various themes. In the following sections, we will delve into the specific themes and codes, providing background information on their origins and significance.

4.1 Part 1: Business Process Embedding

In our SLR, we identified various business processes and related aspects that are impacted by inner source and its measurement. Figure 4 shows the resulting themes (A, B, C) and codes.

4.1.1 Theme A: Management Processes. The first important domain influenced by inner source are management processes which need to be adopted to fit the cross-boundary collaboration pattern of inner source (see Section 5).

Personnel management. With the introduction of inner source, companies must carefully examine Human Resources (HR) processes. One change brought by inner source is the flexible way in which projects are organized and collaboration is facilitated. Rather than fixed projects, developers are now more flexibly assigned [13], which consequently affects HR processes (e.g., recruiting and workforce planning) and not only project management itself. Performance management is also impacted, as middle management often fears losing personal performance goals when making contributions to other organizations [55].

Product management. Managing product development with inner source can be challenging because contributions come from a large number of teams and departments, making it difficult to identify which department contributes to which extent [55]. This lack of clarity can make it harder for product management to track metrics accurately. Precise product management is crucial to managing a product throughout its life-cycle [21]. Inner source can also affect strategic product planning decisions, such as the introduction of software platforms [55]. Therefore, management needs to understand how inner source affects their product management processes and how to design their product within the new development environment to derive maximum benefits from it.

Project management. Cross-boundary collaboration patterns, such as inner source, have a significant impact on project management [31, 34, 65]. As contributions are made across organizational boundaries, companies must adapt their project planning and monitoring processes accordingly. Additionally, involving different legal units alters a project's risk management approach [22, 39, 57]. To account for the impact of inner source, key performance indicators (KPIs) [3, 16] and the Goal-Question-Metric (GQM) principle [6] are often used. It is crucial to ensure that these metrics and processes remain up-to-date when introducing inner source.

4.1.2 Theme B: Accounting Processes. Besides management tasks various accounting processes are also impacted by inner source.

Transfer pricing. One important accounting process is the calculation of the value of IP contributions that flow across legal boundaries, also known as transfer pricing, which is used in taxation. The OECD has defined various well-established methods for calculating such a price [43, 49, 51, 53]. However, when using inner source, the choice of method is still unclear since inner source contributions frequently cross organizational boundaries [10, 48]. Organizations such as the OECD and United Nations (UN) have recognized new software development methods and their potential use as problem domains regarding profit shifting [49, 50, 63].

This problem was not only by companies and tax officials but also within the context of inner source businesses [30]. Researchers have already begun to develop initial approaches to solve this problem [10, 15, 31].

Profit calculation. Introducing inner source development can make accurately assigning value to individual contributing organizations challenging [10, 58], complicating profit calculation from an accounting perspective. Accurate profit and cost calculation are crucial for optimizing operational processes [62] and making informed strategic decisions. As companies adopt inner source, it's essential to adapt profit calculation processes to account for this new dynamic.

Cost calculation. Calculating costs for single departments, or cost centers, is a well-established task. However, in inner source, where code flows across organizational boundaries, it becomes more challenging to assign development costs to individual cost centers using traditional cost calculation approaches such as absorption costing [5].

Previous research [10, 11, 31] has proposed measurement and calculation methods to address this issue in inner source. However, a complete solution has yet to be presented.

Cost estimation. Predicting the future cost evolution is a crucial task in cost accounting, and it is influenced by inner source. Cost estimation plays a vital role in various steps throughout the product's life cycle [7], including maintenance [67]. Many papers have conducted cost estimation within agile environments, such as those by Bilgaiyan et al. [7], Karna et al. [36], and Usman et al. [64]. However, these models are not sufficient for use within inner source (as discussed in Section 5), and further research is necessary to adapt them accordingly.

Accounting for software development. In addition to identifying the general accounting processes that are influenced by inner source, we have also found that there are some initial approaches on how to account for software engineering in this context. The solutions we have come across include general approaches [4], dedicated inner source approaches [31], and those for platform organizations [38] where inner source plays a significant role. However, to date, no complete solution for accounting in inner source has been presented, and further work is needed in this area.

4.1.3 Theme C: Development Processes. Next to dedicated management and accounting processes, we also identified aspects related to the software development process that are worth mentioning.

Cross-boundary collaboration. Inner source involves not only the artifacts produced by software projects, but also the way in which workflows and processes are organized to enable collaboration across organizational and legal boundaries [11, 23, 55, 59, 60]. As such, inner source becomes deeply integrated into the organizational structure of a company, not just its development teams.

It is important to note that not all traditional organizational forms (such as functional, divisional, or matrix structures) are ideal for developing cross-boundary projects. In fact, the functional organization can even be detrimental to software development [28], while a matrix organization may have limitations when it comes to large-scale organizations [25].

There are some existing organizational forms that are designed to support the development of complex products or systems, such as project-based or platform-based organizations [32, 40, 55]. However, there is still no ideal solution for organizing large-scale, high-frequency development work like inner source, even with the use of agile development methodologies [42] and existing best practices and guidelines (such as those outlined by Smite et al. in 2017 [68]).

Development practices. Inner source is deeply integrated into businesses' software development practices, largely due to the high-frequency, peer-review aspect that is adopted from open-source development [13, 24, 58]. Adopting inner source requires companies to rethink their code review processes, how they organize documentation, and how they handle contributors [59]. Inner source also impacts engineering organization and processes in general, such as DevOps [66].

Community building. Inner source requires new processes to be introduced, particularly for community building. To be successful, inner source relies on building communities within the company, which can be achieved through new exchange platforms [59]. It is also important for companies to exchange best practices with other companies that are executing inner source, such as through the InnerSource Commons [2]. Incentivization schemes [13, 18, 59] and related processes are essential for building communities and leveraging inner source within the company.

4.1.4 Concluding Business Process Perspective. In Part 1, we have demonstrated that inner source has an impact on various business processes, including management, accounting, and community building. Inner source necessitates the introduction of new development processes, and as a result, established processes such as personnel, project, and product management need to be adjusted to maintain their effectiveness. Similarly, accounting processes, such as transfer pricing, profit calculations, and cost estimation, require modifications to adapt to the inner source model. Unfortunately, there is a lack of tools and principles for accurate accounting in inner source.

One of the key challenges we identified is the potential for inaccurate cost and profit calculations due to the cross-boundary collaboration pattern of inner source. These calculations are crucial for many strategic and operational decision-making processes within companies. The importance of cost calculation in business is emphasized by the International Federation of Accountants (IFAC):

> "Costing is inextricably linked to the organization's flow of resources to produce goods and services. The more accurately a costing model or system represents the operational flow of resources within an organization, the more clarity decision-makers will have in using cost data." [62]

The accurate measurement of cost and profit flow between organizational units is essential to fully realize the benefits that inner source offers.

4.2 Part 2: Computational Tools and Techniques Embedding

Part 1 provided an overview of how the introduction of inner source can impact business processes. In Part 2, we will examine existing computational tools and techniques that businesses can use for economic assessment.

4.2.1 General Aspects. The purpose of this article is not to provide a comprehensive review and classification of all software effort estimation algorithms, tools, methods, and models. Instead, the aim is to identify and categorize computational tools and techniques that can be generally applied to inner source software development and can help solve business process-related challenges.

We classify a tool or technique as generally suitable for solving inner source challenges if it involves little to no manual work and is, therefore, computable and reproducible. We exclude manual work because certain business processes, such as taxation, require reproducible and extensively documented decision-making. Moreover, including non-computable tools and techniques is not a suitable option for inner source due to its high-frequency nature (code contributions are made by the minute). To be suitable for our analysis, the tools and techniques must be able to identify individual contributions or evaluate cross-boundary collaborations, as most of the problems with existing business processes stem from inner source characteristics.

It's important to note that this article is not an accurate estimation of which topics are being researched more frequently or recently. Instead, we focus on the general algorithmic applicability to the inner source pattern. In addition, we take into consideration reviews, as they present the most important findings concisely.

Many of the reviews we identified focus on a specific methods, technologies, or approaches such as agile development or neural networks. We conducted a thorough review of relevant tools and techniques until further reviews did not yield any new insights related to measuring inner source.

Figure 5 outlines the themes (D, E, F, G) and corresponding codes identified during our thematic analysis. Detailed background information on the origin and motivation of this coding will be presented in the following sections. Table 2 maps the identified papers to their codes and themes, in addition to Figure 5. While some papers stated their approach, others required deeper analysis to identify their methods or classifications. For instance, some approaches were found through literature reviews, while others required examination of the calculation goal.

4.2.2 Theme D: Computation Goals. Our research revealed that existing tools and techniques have been developed to serve various computational objectives. Some are specifically designed for management purposes, while others prioritize historical or predictive calculations. It is important to classify these tools and techniques in order to determine their suitability for the different types of business processes discussed earlier. A detailed discussion of this classification will be presented in Section 5.

Measuring for management. Many of the tools and techniques designed for software engineering are intended for various management tasks, such as calculating frequently used measurements like KPIs [3, 16], implementing the GQM model [6], or determining the contribution of developers to development [29]. In inner source, management metrics can be used to address the fear of middle managers not meeting their performance goals, as previously discussed [55].

Measuring cost/effort. A wide range of tools exists for measuring effort in software engineering, serving various purposes such as optimizing production processes [4], monitoring development, identifying bottlenecks, and future planning [29]. The focus of these tools is often on calculating

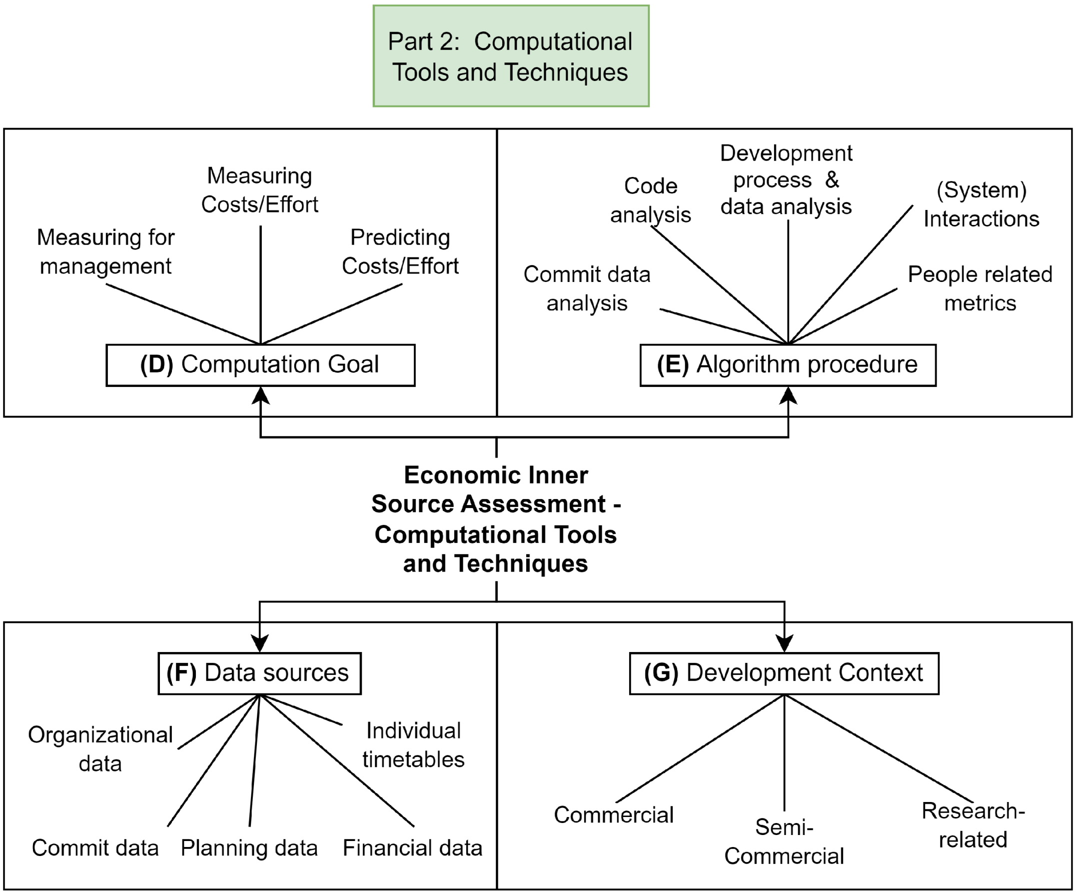

Fig. 5. Themes and codes on the computational tools and techniques perspective of measuring inner source.

historic data, and different approaches have been used to achieve this goal, such as analyzing system interactions [29] or classifying work-time (part-time/full-time) [56].

Predicting cost/effort. We also identified tools and techniques designed to make predictions regarding software development cost and effort. While some of these tools have a more general focus [36], others were developed for specific purposes, such as maintenance [67], agile development [7, 64], or for use with open-source software [54]. Some of these tools and techniques have a long history, dating back to the 1980s with the development of one of the first cost estimation models, COCOMO [8].

4.2.3 Theme E: Algorithmic Procedure. The tools and techniques used in the literature encompass a wide range of procedures to conduct their calculations. Many of the identified articles relied on intuitive calculations for aspects that are easy to quantify, while allocating less emphasis on quantifying social impact factors. In the following sections, we will delve into the suitability of specific approaches for their application within inner source.

Code analysis. One possible method for assessing inner source is analyzing the written code, which can involve either the code currently under development (code committed) [4] or more general metrics based on the final product's source code [67].

Commit data analysis. In addition, it is possible to analyze the commit data beyond the code that is generated during the development process. The approaches we identified are mostly based on mathematical calculations using lines of code and timestamps [10, 47]. This approach is especially

Table 2. Mapping between Papers and the Identified Aspects (Codes and Themes)

		Computation Goal			**Algorithmic procedure**							**Data sources**				
Paper	Type	Management measuring	Measuring Effort	Predicting Effort	Code analysis	Commit analysis	People related metrics	System interactions	Development process and data analysis	Machine Learning	Mathematical approach used	Organisational data	Commit data	Planning data	Financial data	Time tables
[7]	R			x						x	x			x	x	
[35]	A		x								x			x		
[67]	R			x	x	x	x	x	x		x		x			
[36]	A			x					x		x			x		
[47]	A		x			x	x				x		x			
[4]	A		x		x			x					x		x	x
[56]	A		x			x	x				x		x			
[54]	A			x			x				x	x		x		
[29]	A	x	x					x					x			
[20]	T					x			x			x	x			x
[3]	M	x				x			x				x	x	x	
[16]	M	x				x	x		x			x	x	x	x	x
[6]	M	x					x		x			x		x	x	x
[64]	R			x		x	x			x	x	x	x	x		
[8]	A			x							x			x		
[10]	A		x			x					x	x	x		x	

A = Algorithm; M = Metric; R = Review, T = Toolset.

suitable for inner source development and has already been applied to it [10], as commit data represents the lowest logical level of IP contributions made.

People-related metrics. We also identified the use of people-related information as useful for estimating inner source. This includes information such as whether a person is part- or full-time [47] or the developer's experience [54]. This information may be usable for people-related management processes, such as within KPI/GQM calculations [3, 6, 16].

(System) interactions. Another method is to utilize interactions with different internal systems within businesses [4, 29] and base further calculations on the measured interactions, sometimes called activity-based [67].

Development process and data analysis. Moreover, it is possible to base measurements on insights into the development process itself, such as sprints [36]. This information can also be integrated into tools such as GrimoireLab [20], which is already used for analyzing inner source. Additionally, management calculations often rely on measuring these processes.

It is worth mentioning that a wide variety of mathematical approaches were used in the studies reviewed. Linear regressions were employed in most cases (e.g., [10, 47]) to varying degrees, and statistical analyses were also utilized (e.g., [10, 35, 56, 67]). These papers were additionally highlighted in Table 2 to indicate the use of mathematical methods.

We will not look closer into the vast array of machine learning algorithms and concepts available, as they are typically developed for specific purposes, and their transferability must be evaluated independently in each case.

4.2.4 Theme F: Data Sources. In our review, we found suitable data sources for economic assessments of inner source. These sources can be particularly useful for newcomers to inner source measurement, providing an overview of the necessary data for accurate measurement.

Organizational data. Organizational data is crucial for many business processes that are influenced by the cross-boundary collaboration pattern of inner source (see Part 1). Having accurate and easily accessible organizational data helps companies adjust their processes to fit the inner source paradigm, thereby avoiding unnecessary risks. Such data can include the company's structures, such as teams and hierarchies [16], as well as employee numbers [20].

Commit data. Commit data is a vital source of information for measuring software development, as a large number of tools and techniques rely on it. Therefore, commit data is necessary for measuring inner source development. Commit data is one of the most critical data sources for inner source assessment, and it has already been used in dedicated inner source measurement research [10, 12, 31].

Planning data. Some identified papers have used planning data related to software projects, such as start and end dates [3], use-cases for analyzing functions and story-points [35], or sprint data [36]. However, these have not played a major role in the research dedicated to inner source that we identified.

Financial data. Although it is not often explicitly mentioned, financial data is an essential data source. As many of the tools and techniques directly or indirectly calculate costs or profits (e.g., [10]), the use of financial data is necessary.

Individual timetables. Our review also showed that data from various software systems used within companies can prove useful in inner source assessment. We identified techniques that rely on the time worked by individual people [6, 16, 47] or that require such data as additional input [4]. The Grimoirelabs tool [20] integrated meeting and communication data (Slack, Telegram, and E-Mails) as well as data from ticket systems (like Jira) for their analysis.

To summarize, the various data sources we presented can be used individually or in combination to perform comprehensive inner-source analyses and adapt business processes accordingly. While some dedicated measurement approaches for inner source have utilized single sources, a comprehensive tool has not yet been presented.

4.2.5 Theme G: Development Context. Although it plays a minor role in addressing inner source measurement challenges, the context in which the tools and techniques were developed is a noteworthy aspect that we were able to identify.

Commercial. Some tools and techniques (which we have reviewed to a lesser extent) were developed within a commercial environment. For example, Price System developed a parametric cost estimation system for hardware development in the 1960s and 70s [61]. However, since commercial systems are usually not freely available, they are not the focus of our research.

Semi-Commercial. Other tools and techniques were developed in a commercial context but are publicly available through research publications or books. Examples include the Cocomo model [8] and (Wideband) Delphi [7, 8].

Research-Related/Non-Commercial. The majority of the reviewed papers (not previously mentioned) originate from research or non-commercial contexts, although they might be used in commercial contexts.

5 PART 3: THEMATIC MAP

This section brings together Parts 1 and 2, identifying the relationship between the business processes affected by inner source and existing measurement approaches for software development. Part 3 is split into three sub-parts (3a, 3b, 3c), with 3a providing an overview and 3b and 3c analyzing the most important relationships and implications in-depth.

5.1 Part 3a: Thematic Map

In the following, we will present the thematic map created within the research process as proposed by Braun and Clarke within Step 3 of the thematic analysis (See Figure 2). Braun and Clarke suggest creating a mind map (also called a thematic map) during the research. Consequently, all the connections between the themes and codes and the following in-depth discussions are based on the already presented results of the SLR and related literature (Section 5).

Figure 6 presents a high-level thematic map that depicts the themes in the two review domains (business processes, tools, and techniques) and the general connections between them. In the following sections (Part 3b and 3c—Sections 5.2 and 5.3), we will examine these themes in more detail. Since not all themes are equally related to each other, we have categorized the connections into different types to make them easier to understand.

Close connections: Themes that are closely connected share codes that are somehow related to each other. This means that these themes have a similar way of working, solve similar problems, or have similar logic behind their classification. We have identified two main connections between the themes. Firstly, all business processes are related to the goals of the identified tools. Secondly, the way the tools and procedures work is closely related to their goals, as these tools and techniques are designed to solve problems that occur in similar business situations.

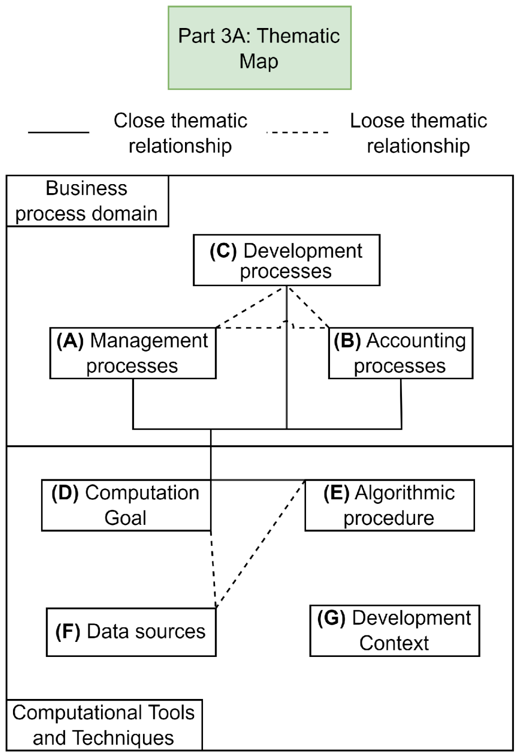

Fig. 6. Thematic map showing theme relationships.

Loose connections: Themes with a loose connection are related to each other, but their processes, tools, and techniques do not impact each other in a centrally important way for the economic inner source assessment. For example, the computational goals (Theme D) and algorithmic procedures (Theme E) of tools and techniques are influenced by the available data sources (Theme F), but are not of central importance to them with regards to inner source measurement. The same is true for the business processes influenced by inner source (Themes A to C).

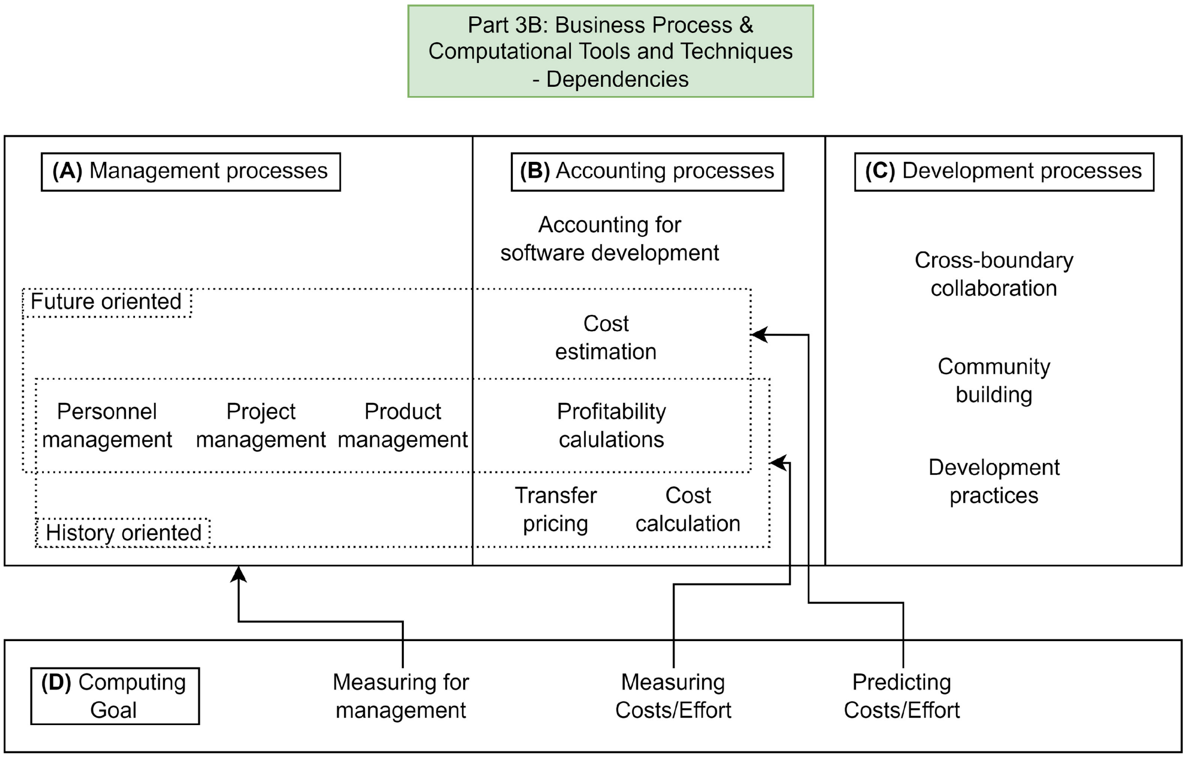

Fig. 7. Overview over the connections between the codes connecting the business and algorithmic view.

No connections: Theme G has no connection to any of the other themes. This is because the development context of a tool or technique (if it is a commercial product or a research article) does not play an important role in solving the inner source business measurement problems.

In particular, themes that are closely related are of particular interest in this article. They are discussed in more detail in the following sections.

5.2 Part 3b: Business Process and Tools and Techniques Dependencies

The following section details the link between the identified business processes and tools and techniques. Figure 7 displays four boxes, with the top three representing the business processes impacted by inner source (Themes A to C), and the bottom representing the objectives of the tools and techniques analyzed (Theme D). The figure illustrates the interconnection between the themes in both domains.

5.2.1 Dependency Analysis. After reviewing the literature, it became apparent that the goals of computational tools and techniques (Theme D) cannot be entirely separated from the business processes affected by inner source (Themes A to C). This is due to the fact that the majority of the analyzed tools and techniques were originally designed to serve specific business purposes, such as calculating maintenance costs or estimating work time. As a result, these tools and techniques are intertwined with the business processes that they support.

Figure 7 illustrates that some processes in software development focus exclusively on either historic calculations (e.g., transfer pricing, cost calculation, Theme B) or estimations/predictions (e.g., cost estimations, Theme B), while others require both (e.g., profitability calculations in Theme B, or personnel management, product management, and project management in Theme A).

Tools and techniques aimed at predicting cost or effort (Theme D, see Table 2) are best suited for prediction-oriented processes, such as management processes and profit/cost estimations in accounting. Conversely, tools and techniques designed to measure historic cost or effort are better suited for history-oriented business processes, like cost calculation and transfer pricing.

Additionally, tools and techniques classified under the “Measuring for management” code (Theme D) are well-aligned with the management-related processes of inner source (Theme A). For example, KPI calculations are more suitable for addressing the project management challenges of inner source.

These findings highlight the importance of measuring software development, particularly in the context of inner source. Inner source development impacts a wide range of business processes (Theme A to C), and being able to accurately measure it using the proposed tools and techniques (Themes D to G) can help address various challenges associated with inner source development.

5.2.2 Business and Research Implications. After conducting a thematic analysis, we found that inner source has a significant impact on various strategic and operational business processes beyond software development, such as accounting (see Theme B) and management (see Theme A). To effectively implement inner source, new community building and incentivization processes need to be introduced (see Theme C).

While several tools and techniques have been designed to support traditional development environments (see Theme D), not all of them were originally intended for inner source. Many were created for predictive or history-oriented business processes outside of inner source. Therefore, future research on inner source measurement should focus on making predictions and calculating historic events specifically with inner source in mind, in order to comprehensively handle it.

To provide guidance for future inner source measurement tools or models, we need to further examine the codes and themes in our analysis to identify which tools and techniques are better suited for the inner source paradigm, and which may require significant adjustments by future researchers. This will help create a comprehensive inner source measurement tool or model that can assist with as many business processes as possible.

5.3 Part 3c: Computational Tools and Techniques Usability Analysis

This section examines the applicability of certain tools and techniques for the inner source domain by analyzing their procedures (Theme E) and goals (Theme D).

Figure 8 illustrates the relationship between Themes D and E in detail. The figure consists of two main boxes, with Theme D (Gray) representing the computational goals and Theme E (White) representing the algorithmic procedures used. The figure depicts the identified procedures, classified by their suitability for specific calculation purposes, and how they relate to the computational goals of the algorithm. It provides an overview of the suitability of tools and techniques for applying certain procedures in inner source calculations or predictions.

Although we initially selected tools and techniques that are generally applicable to cross-boundary collaboration, not all of them are equally suitable for use in inner source or in all situations that come with it (e.g., predictive vs historic calculations discussed in Section 5.2). We have classified these tools and techniques into three basic types:

(1) Well-suited approaches: These tools and techniques are easily applicable to the cross-boundary pattern of inner source.
(2) Supportive approaches/approaches suitable with minor adaptations: These tools and techniques can be used in inner source with minor adjustments or provide additional support for other tools and techniques.
(3) Approaches suitable with major adaptations: These tools and techniques may be generally applicable to inner source but require major adjustments to benefit inner source.

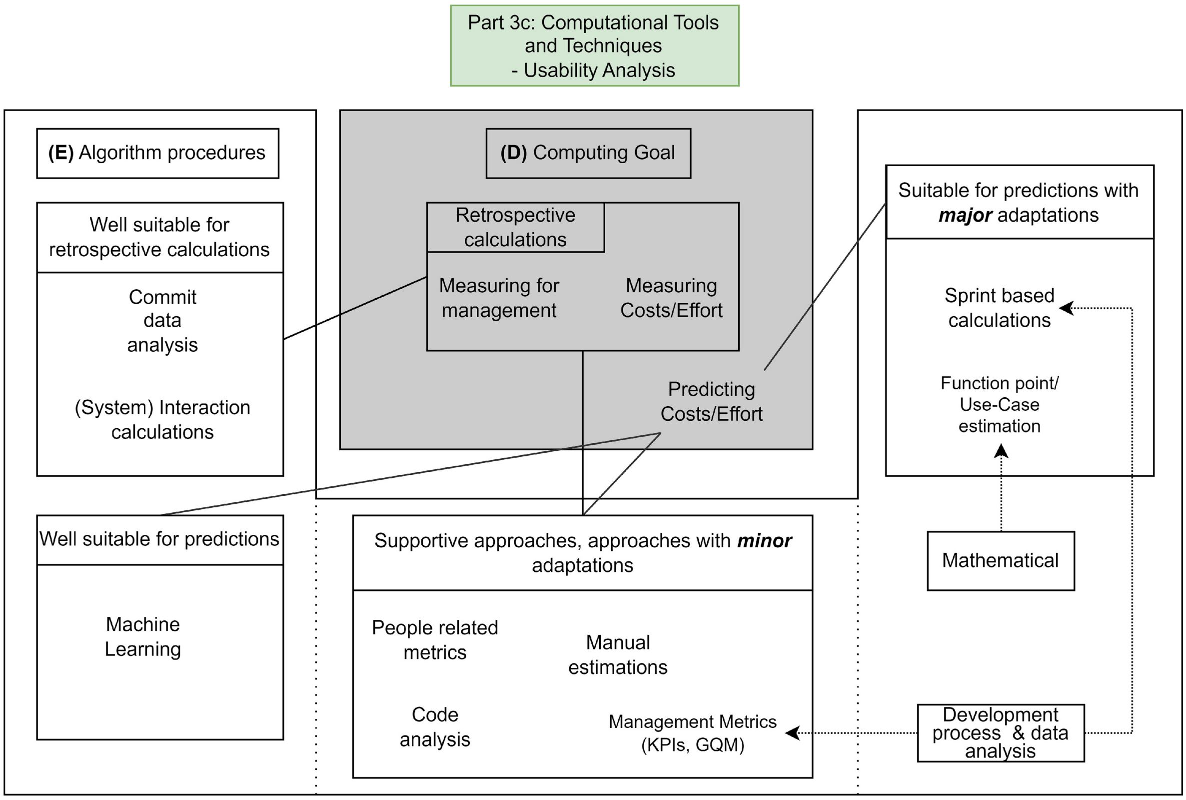

Fig. 8. Overview over the connections between the goals and procedures of computational tools and techniques.

Our main focus is on the ease of assessing individual cross-boundary contributions (transfers) with the identified tools and techniques, as these represent the most elementary part of inner source development.

5.3.1 Inner Source Usability Analysis.

Well-suited approaches. Tools and techniques in this category are easy to apply to the cross-boundary pattern of inner source. Some are well-suited for retrospective calculations, while others are well-suited for predictive calculations.

For retrospective calculations using historic data, tools, and techniques that support cross-boundary collaboration are ideal. Examples include individual system interactions [4, 29] and commit data [10]. These tools are directly assignable to contributions and are easy to use in management and accounting processes presented with Themes A and B.

To make predictions, a dedicated machine-learning model might be suitable, although specific machine learning solutions were not the focus of this review. The applicability of machine learning heavily depends on the goal and the business environment.

Supportive approaches. Tools and techniques in this group are usable for other tools and techniques as input or are suitable after minor adaptations.

Various tools and techniques fit into this category. Source code analysis, for example, may not be directly related to the transferred intellectual property, but it can enable code metrics (e.g., code complexity) on a commit level [4, 67]. The same is true for people-related metrics such as full- and part-time handling [56]. Moreover, management metrics and methods like KPI calculations can provide value support and validation information for potential future inner source tools and techniques developed in research and business [3, 6, 16].

Although manual methods were not actively reviewed, in this article, some of the methods mentioned in the literature, such as expert judgments, planning poker, and the Delphi method [7], are suitable for validation purposes of future tools. However, using manual inputs should be the exception, as it is time-intensive for the large amount of high-frequency inner source contributions and not easy to include in future software tools.

Approaches suitable with major adaptations. Tools and techniques in this category require significant modifications to be suitable for the inner source paradigm. Within this group, we have identified only predictive algorithms.

For example, function point/Use-case estimation, often used for effort estimation [35], does not always capture the fine-grained level of flexibility required for inner source contributions. Further research is needed to investigate how these methods can be adapted to meet the needs of inner source development. Similarly, sprint-based calculations (e.g., [36]) do not directly correlate with inner source IP transfer, but may provide valuable insights after appropriate adjustments have been made.

5.3.2 Business and Research Implications. We classified several tools and techniques and found that most of them require significant adjustment to be suitable for use in inner source, while only a few are well-suited for economic assessment in inner source. The supportive approaches that are well-suited for inner source deal with data structures that are directly assignable to transferable work within inner source, such as commits and system interactions.

One key takeaway from our classification is that most of the existing tools and techniques were not developed with inner source in mind, and therefore, require major or minor adaptations for use in inner source. Furthermore, the tools and techniques that are applicable to inner source were developed for specific use cases, such as transfer pricing [10].

To address these issues, we propose future research to integrate all suitable approaches into a tool that is specifically designed for the economic assessment of inner source. Such a tool would enable companies to adjust their processes and take advantage of all the benefits that inner source has to offer.

5.4 Key Findings

The goal of this article was to identify the economic impact of inner source on businesses and their processes, and to determine how such an impact can be quantified. Through thematic analysis and the resulting thematic map, we are able to answer our initially proposed research question.

We discovered that inner source affects a wide range of business processes within management and accounting (Part 1, Themes A to C). We also found that existing software development practices, such as code review and documentation handling, need to be adapted to facilitate inner source. Furthermore, new processes for community building, such as inner source incentivization, need to be implemented to fully capitalize on the benefits of inner source.

Regarding quantification, we found that although many computational tools and practices exist to measure software development and support existing business processes (Part 2, Themes D to G), most are not yet suitable for handling the cross-organizational collaboration patterns of inner source (Part 3). We identified preliminary data sources and procedures capable of handling cross-organizational IP contributions that are well-suited for measuring inner source and related business processes.

As a result, future research should focus on developing tools and techniques that are capable of handling inner source flows and applying them within businesses. Additionally, we identified the need for both predictive and retrospective calculations to comprehensively cover inner source measurement.

6 OUTLOOK AND CONCLUSION

6.1 Limitations

In this section, we will discuss the limitations of our findings using the trustworthiness criteria proposed by Lincoln and Guba [41]. These criteria include credibility, confirmability, transferability, and dependability.

Credibility refers to whether the findings reflect the reality. In our work, we limited our review to literature sources only, as described in our review protocol in Section 3.3. While we did not directly include findings from industry through interviews or case studies, we mitigated this limitation by carefully selecting papers dealing with case studies or reviews handling industry perspectives and feedback. By doing so, we were able to integrate multiple industry perspectives in a thoroughly evaluated manner, as we checked the quality criteria of the papers and assessed their practical relevance.

Confirmability refers to avoiding researcher bias. We ensured confirmability in our review by having both authors conduct an independent review and thematic analysis and agreeing on the findings afterward (inter-rater reliability). Additionally, we only included peer-reviewed papers and other literature reviews to reduce the risk of researcher bias.

Transferability refers to the applicability of the findings outside of the article scope. We recognize that only a few papers we identified have directly taken inner source into consideration, as research in that domain is not yet widespread enough. However, we addressed this limitation by considering articles outside the inner source domain and reviewing the applicability of the identified tools and techniques to the inner source domain, as presented in Section 5. Moreover, we limited our literature review to reproducible tools and techniques (e.g., no machine learning) that involve (almost) no manual work. This enabled the transferability of the findings to a wide range of business processes and development measurement domains outside the review scope of inner source.

Finally, dependability refers to the replicability of the study design. While our chosen study design may limit dependability, we provided all the information proposed by Kitchenham [37] to ensure replicability. This includes a detailed review protocol, executed steps, key words, and quality criteria. We also provided the thematic maps created throughout our research process, following Braun and Clarke's [9] thematic analysis, to ensure transparency.

6.2 Broader Research Influence

The economic assessment of inner source provides a foundation for addressing the research questions posed by Edison et al. [23]. They identified a lack of clarity regarding how improvements in management through inner source can be measured, and our review has shown that tools and techniques capable of performing predictive calculations are essential for managing inner source-related processes.

Additionally, Edison et al. stated that the creation of business value through inner source is unclear. Our research lays the groundwork for answering this question by providing an overview of how inner source metrics should be created to comprehensively measure its impact.

Moreover, economic assessments of inner source can be crucial tools for a wide range of research agendas based on economic implications, extending beyond inner source development to accelerate general economic and software-related research.

6.3 Future Research Propositions

Based on the results of our SLR and thematic analysis, we propose several aspects for future research. First, future research should focus on developing comprehensive tools and techniques to assess the impact of inner source on businesses and their processes. Such tools should integrate a wide range of data sources, including system and process data, in addition to commit data as used in current solutions (e.g., Buchner et al., 2022).

Another important challenge is to quantify the social aspects of inner source development, which can provide valuable insights into inner source adoption and team dynamics. To address this, further research should extend existing work on inner source community building and incentivization, and measure their impact.

In addition to retrospective tools and techniques, future research should also explore predictive algorithms dedicated to inner source development. This would help improve inner source planning and control, and provide valuable insights and improvements for both research and business through the use of machine learning algorithms.

To facilitate the introduction and management of inner source, we propose building one or more inner source measurement and accounting tools that can comprehensively deliver all necessary information. These tools should also be able to easily adapt existing processes to this new paradigm.

To measure the impact of inner source on businesses and their processes, we recommend conducting case studies that evaluate the usability of inner source measurement tools. We found that inner source influences a company's way of organizing their development teams and overall organizational structure (e.g., functional vs. platform organization, as discussed in Part 1). Therefore, we propose further research to measure the performance of different organizational structures and compare them with the application of inner source in terms of efficiency.

6.4 Conclusions

In this review, we analyze two perspectives on assessing economic benefits of inner source: the business process perspective (Part 1) and the tools and techniques perspective (Part 2). We then explore how these tools and techniques align with the business processes influenced by inner source, particularly for management and accounting. Additionally, we conduct an analysis of which types of tools and techniques are well-suited to handle business processes affected by inner source, and identify areas that require further research.

Our key finding is that current tools and techniques are insufficient to provide a comprehensive assessment of the economic benefits of inner source. Existing tools and techniques that are applicable to inner source rely on data sources and procedures that can identify cross-boundary IP flow. Future research should focus on developing predictive and retrospective directed processes that can handle these assessments more effectively.

Overall, this SLR lays the foundation for potential future research that can improve inner source adoption, making it easier for companies to become more efficient and agile in responding to new market needs.COMP: Please move the funding information to the first page footnote.

ACKNOWLEDGMENTS

We thank our colleagues for continuous feedback and proofreading of this article. We acknowledge the use of ChatGPT (GPT-3.5, Version 23rd March 2023) for improving the readability of this research paper (and only that). As non-native speakers, we utilized it solely for correcting grammatical errors, correcting and suggesting appropriate words and phrases, and detecting spelling mistakes. We copied and pasted each paragraph of our research paper individually into the software. We then thoroughly reviewed each correction in every paragraph suggested by ChatGPT to ensure that our original intention remained unchanged, and that no contents were added or deleted. This process ensured that the improvements suggestions did not alter the meaning of

our work in any way. ChatGPT was not used in any way for the literature identification, content creation, and writing of that paper.

REFERENCES

[1] InnerSource Commons. 2021. State of InnerSource 2021. Retrieved July 12, 2023 from https://innersourcecommons.org/learn/research/state-of-innersource-survey-2021/
[2] InnerSource Commons. 2023. Community. Retrieved March 21, 2023 from https://innersourcecommons.org/community/
[3] Ž. Antolić. 2008. An example of using key performance indicators for software development process efficiency evaluation. *R&D Center Ericsson Nikola Tesla* 6 (2008), 1–6.
[4] Saulius Astromskis, Andrea Janes, Alberto Sillitti, and Giancarlo Succi. 2014. An approach to non-invasive cost accounting. In *Proceedings of the 2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications*. IEEE, 30–37. DOI: https://doi.org/10.1109/SEAA.2014.53
[5] Bunea Bontas Cristina Aurora. 2013. The cost of production under direct costing and absorption costing—a comparative approach. *Annals—Economy Series* 2 (2013), 123–129.
[6] Victor Basili, Mikael Lindvall, Myrna Regardie, Carolyn Seaman, Jens Heidrich, Jürgen Münch, Dieter Rombach, and Adam Trendowicz. 2010. Linking software development and business strategy through measurement. *Computer* 43, 4 (2010), 57–65. DOI: https://doi.org/10.1109/MC.2010.108
[7] Saurabh Bilgaiyan, Santwana Sagnika, Samaresh Mishra, and M. N. Das. 2017. A systematic review on software cost estimation in agile software development. *Journal of Engineering Science and Technology Review* 10, 4 (2017), 51–64. DOI: https://doi.org/10.25103/jestr.104.08
[8] Barry W. Boehm. 1984. Software engineering economics. *IEEE Transactions on Software Engineering* SE-10, 1 (1984), 4–21. DOI: https://doi.org/10.1109/TSE.1984.5010193
[9] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (2006), 77–101. DOI: https://doi.org/10.1191/1478088706qp063oa
[10] Stefan Buchner and Dirk Riehle. 2022. Calculating the costs of inner source collaboration by computing the time worked. In *Proceedings of the 55th Hawaii International Conference on System Sciences, HICSS'22*. virtual.
[11] Maximilian Capraro. 2020. *Measuring Inner Source Collaboration*. Ph. D. Dissertation. Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU).
[12] Maximilian Capraro, Michael Dorner, and Dirk Riehle. 2018. The patch-flow method for measuring inner source collaboration. In *Proceedings of the 15th International Conference on Mining Software Repositories (MSR'18)*. Association for Computing Machinery, New York, NY, 515–525. DOI: https://doi.org/10.1145/3196398.3196417
[13] Maximilian Capraro and Dirk Riehle. 2016. Inner source definition, benefits, and challenges. *ACM Computing Surveys* 49, 4, Article 67 (Dec. 2016), 36 pages. DOI: https://doi.org/10.1145/2856821
[14] Noel Carroll, Lorraine Morgan, and Kieran Conboy. 2018. Examining the impact of adopting inner source software practices. In *Proceedings of the 14th International Symposium on Open Collaboration (OpenSym'18)*. Association for Computing Machinery, New York, NY, Article 6, 7 pages. DOI: https://doi.org/10.1145/3233391.3233530
[15] Yasin Ceran, Milind Dawande, Dengpan Liu, and Vijay Mookerjee. 2014. Optimal software reuse in incremental software development: A transfer pricing approach. *Management Science* 60, 3 (2014), 541–559. DOI: https://doi.org/10.1287/mnsc.2013.1757
[16] Tjan-Hien Cheng, Slinger Jansen, and Marc Remmers. 2009. Controlling and monitoring agile software development in three dutch product software companies. In *Proceedings of the 2009 ICSE Workshop on Software Development Governance*. IEEE, 29–35. DOI: https://doi.org/10.1109/SDG.2009.5071334
[17] Danese Cooper and Klaas-Jan Stol. 2018. *Adopting InnerSource: Principles and Case Studies*. O'Reilly Media.
[18] Tapajit Dey, Willem Jiang, and Brian Fitzgerald. 2022. Knights and gold stars: A tale of innersource incentivization. *IEEE Software* 39, 6 (2022), 88–98. DOI: https://doi.org/10.1109/MS.2022.3192647
[19] Isabel Drost-Fromm. 2018. Borrowing open source practices at europace. In *Proceedings of the Adopting InnerSource: Principles and Case Studies*. Danese Cooper and Klaas-Jan Stol (Eds.), O'Reilly Media.
[20] Santiago Dueñas, Valerio Cosentino, Jesus M. Gonzalez-Barahona, Alvaro del Castillo San Felix, Daniel Izquierdo-Cortazar, Luis Cañas-Díaz, and Alberto Pérez García-Plaza. 2021. GrimoireLab: A toolset for software development analytics. *PeerJ Computer Science* 7 (July 2021), e601. DOI: https://doi.org/10.7717/peerj-cs.601
[21] Christof Ebert. 2014. Software product management. *IEEE Software* 31, 3 (2014), 21–24. DOI: https://doi.org/10.1109/MS.2014.72
[22] Christof Ebert, Bvs Krishna Murthy, and Namo Narayan Jha. 2008. Managing risks in global software engineering: Principles and practices. In *Proceedings of the 2008 IEEE International Conference on Global Software Engineering*. IEEE, 131–140. DOI: https://doi.org/10.1109/ICGSE.2008.12

[23] Henry Edison, Noel Carroll, Lorraine Morgan, and Kieran Conboy. 2020. Inner source software development: Current thinking and an agenda for future research. *Journal of Systems and Software* 163 (05 2020), 110520. DOI: https://doi.org/10.1016/j.jss.2020.110520

[24] Joseph Feller and Brian Fitzgerald. 2000. A framework analysis of the open source software development paradigm. In *Proceedings of the 21st International Conference on Information Systems (ICIS'00)*. Association for Information Systems, 58–69.

[25] Robert C. Ford and W. Alan Randolph. 1992. Cross-functional structures: A review and integration of matrix organization and project management. *Journal of Management* 18, 2 (1992), 267–294. DOI: https://doi.org/10.1177/014920639201800204

[26] Martin Fowler. 1996. *Analysis Patterns: Reusable Object Models (Addison-Wesley Series in Object-Oriented Software Engineerin)*. Addison-Wesley Longman, Amsterdam.

[27] Thomas Froment and Guillaume Angier de Lohéac. 2021. The convergence of struggles! Reusability assessment of inner-source components for product lines. *INSIGHT* 24, 1 (2021), 30–34. DOI: https://doi.org/10.1002/inst.12325

[28] R. Fuller. 2019. Functional organization of software groups considered harmful. In *Proceedings of the 2019 IEEE/ACM International Conference on Software and System Processes (ICSSP)*. IEEE, 120–124. DOI: https://doi.org/10.1109/ICSSP.2019.00024

[29] Georgios Gousios, Eirini Kalliamvakou, and Diomidis Spinellis. 2008. Measuring developer contribution from software repository data. In *Proceedings of the 2008 International Working Conference on Mining Software Repositories (MSR'08)*. Association for Computing Machinery, New York, NY, 129–132. DOI: https://doi.org/10.1145/1370750.1370781

[30] Georg Gruetter, Diogo Fregonese, and Jason Zink. 2018. Living in a BIOSphere at robert bosch. In *Adopting InnerSource: Principles and Case Studies*. Danese Cooper and Klaas-Jan Stol (Eds.). O'Reilly Media.

[31] Julian Hirsch and Dirk Riehle. 2022. Management accounting concepts for inner source software engineering. In *Lecture Notes in Business Information Processing* (2022-11-08/2022-11-11), Noel Carroll, Anh Nguyen-Duc, Xiaofeng Wang, and Viktoria Stray (Eds.), Vol. 463 LNBIP. Springer Science and Business Media Deutschland GmbH, 101–116. DOI: https://doi.org/10.1007/978-3-031-20706-8_7

[32] Mike Hobday. 2000. The project-based organisation: An ideal form for managing complex products and systems? *Research Policy* 29, 7 (2000), 871–893. DOI: https://doi.org/10.1016/S0048-7333(00)00110-4

[33] Martin Ivarsson and Tony Gorschek. 2011. A method for evaluating rigor and industrial relevance of technology evaluations. *Empirical Software Engineering* 16, 3 (June 2011), 365–395. DOI: https://doi.org/10.1007/s10664-010-9146-4

[34] Capers Jones. 2004. Software project management practices: Failure versus success. *CrossTalk: The Journal of Defense Software Engineering* 17, 10 (2004), 5–9.

[35] Sungjoo Kang, Okjoo Choi, and Jongmoon Baik. 2010. Model-based dynamic cost estimation and tracking method for agile software development. In *Proceedings of the 2010 IEEE/ACIS 9th International Conference on Computer and Information Science*. IEEE, 743–748. DOI: https://doi.org/10.1109/ICIS.2010.126

[36] Hrvoje Karna. 2020. Data mining approach to effort modeling on agile software projects. *Informatica* 44, 2 (06 2020), 231–239. DOI: https://doi.org/10.31449/inf.v44i2.2759

[37] Barbara Kitchenham. 2004. Procedures for performing systematic reviews. *Keele, UK, Keele University* 33, 2004 (2004), 1–26.

[38] Martin Kornberger, Dane Pflueger, and Jan Mouritsen. 2017. Evaluative infrastructures: Accounting for platform organization. *Accounting, Organizations and Society* 60 (2017), 79–95. DOI: https://doi.org/10.1016/j.aos.2017.05.002

[39] Y. H. Kwak and J. Stoddard. 2004. Project risk management: Lessons learned from software development environment. *Technovation* 24, 11 (2004), 915–920. DOI: https://doi.org/10.1016/S0166-4972(03)00033-6

[40] Leonardo Leite, Fabio Kon, Gustavo Pinto, and Paulo Meirelles. 2020. Platform teams: An organizational structure for continuous delivery. In *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops (ICSEW'20)*. Association for Computing Machinery, New York, NY, 505–511. DOI: https://doi.org/10.1145/3387940.3391455

[41] Yvonna S. Lincoln and Egon G. Guba. 1985. *Naturalistic Inquiry*. Sage.

[42] Mikael Lindvall, Dirk Muthig, Aldo Dagnino, Christina Wallin, Michael Stupperich, David Kiefer, John May, and T. Kahkonen. 2005. Agile software development in large organizations. *Computer* 37 (01 2005), 26–34. DOI: https://doi.org/10.1109/MC.2004.231

[43] Orly Mazur. 2016. Transfer pricing challenges in the cloud. *Boston College Law Review* 57, 2 (2016), 643–693. Retrieved from https://lawdigitalcommons.bc.edu/bclr/vol57/iss2/6

[44] William E. McCarthy. 1982. The REA accounting model: A generalized framework for accounting systems in a shared data environment. *The Accounting Review* 57, 3 (1982), 554–578. Retrieved from http://www.jstor.org/stable/246878

[45] Lorraine Morgan, Joseph Feller, and Patrick Finnegan. 2011. Exploring inner source as a form of intraorganisational open innovation. In *19th European Conference on Information Systems, ECIS 2011, Helsinki, Finland, June 9–11, 2011*. Virpi Kristiina Tuunainen, Matti Rossi, and Joe Nandhakumar (Eds.). 151. Retrieved from http://aisel.aisnet.org/ecis2011/151

[46] Lorraine Morgan, Rob Gleasure, Abayomi Baiyere, and Hong Phuc Dang. 2021. Share and share alike: How inner source can help create new digital platforms. *California Management Review* 64, 1 (2021), 90–112. DOI: https://doi.org/10.1177/00081256211044830

[47] Donatien Koulla Moulla, Alain Abran, and Kolyang. 2021. Duration estimation models for open source software projects. *International Journal of Information Technology and Computer Science* 13, 1 (02 2021), 1–17. DOI: https://doi.org/10.5815/ijitcs.2021.01.01

[48] A. Neumann. 2019. *Transfer Pricing in Inner Source Software Development*. Master's thesis. Hochschule des Bundes für öffentliche Verwaltung, Bruhl, Germany.

[49] OECD. 2015. *Aligning Transfer Pricing Outcomes with Value Creation, Actions 8-10-2015 Final Reports*. OECD. 192 pages. DOI: https://doi.org/10.1787/9789264241244-en

[50] OECD. 2017. *OECD Transfer Pricing Guidelines for Multinational Enterprises and Tax Administrations 2017*. OECD. 608 pages. DOI: https://doi.org/10.1787/tpg-2017-en

[51] Marcel Olbert and Christoph Spengel. 2017. International taxation in the digital economy : challenge accepted? *World Tax Journal : WTJ* 9, 1 (2017), 3–46. Retrieved from https://madoc.bib.uni-mannheim.de/41867/

[52] Open Source Initiative. 2007. *The Open Source Definition*. Retrieved November 03, 2021 from https://opensource.org/osd

[53] Christian Plesner Rossing, Martine Cools, and Carsten Rohde. 2017. International transfer pricing in multinational enterprises. *Journal of Accounting Education* 39, C (2017), 55–67. Retrieved from https://EconPapers.repec.org/RePEc:eee:joaced:v:39:y:2017:i:c:p:55-67

[54] Fumin Qi, Xiao-Yuan Jing, Xiaoke Zhu, Xiaoyuan Xie, Baowen Xu, and Shi Ying. 2017. Software effort estimation based on open source projects: Case study of Github. *Information and Software Technology* 92 (2017), 145–157. DOI: https://doi.org/10.1016/j.infsof.2017.07.015

[55] Dirk Riehle, Maximilian Capraro, Detlef Kips, and Lars Horn. 2016. Inner source in platform-based product engineering. *IEEE Transactions on Software Engineering* 42, 12 (12 2016), 1162–1177. DOI: https://doi.org/10.1109/TSE.2016.2554553

[56] Gregorio Robles, Jesús M. González-Barahona, Carlos Cervigón, Andrea Capiluppi, and Daniel Izquierdo-Cortázar. 2014. Estimating development effort in free/open source software projects by mining software repositories: A case study of OpenStack. In *Proceedings of the 11th Working Conference on Mining Software Repositories (MSR 2014)*. Association for Computing Machinery, New York, NY, 222–231. DOI: https://doi.org/10.1145/2597073.2597107

[57] Geoffrey G. Roy. 2004. A risk management framework for software engineering practice. In *Proceedings of the 2004 Australian Software Engineering Conference*. IEEE, 60–67. DOI: https://doi.org/10.1109/ASWEC.2004.1290458

[58] Klaas-Jan Stol, Paris Avgeriou, Muhammad Ali Babar, Yan Lucas, and Brian Fitzgerald. 2014. Key factors for adopting inner source. *ACM Transactions on Software Engineering and Methodology* 23, 2, Article 18 (April 2014), 35 pages. DOI: https://doi.org/10.1145/2533685

[59] Klaas-Jan Stol, Muhammad Ali Babar, Paris Avgeriou, and Brian Fitzgerald. 2011. A comparative study of challenges in integrating Open Source Software and Inner Source Software. *Information and Software Technology* 53, 12 (2011), 1319–1336. DOI: https://doi.org/10.1016/j.infsof.2011.06.007

[60] Klaas-Jan Stol and Brian Fitzgerald. 2015. Inner source–adopting open source development practices in organizations: A tutorial. *IEEE Software* 32, 4 (2015), 60–67. DOI: https://doi.org/10.1109/MS.2014.77

[61] PRICE Systems. 2021. Company overview: Price® Systems. Retrieved December 15, 2021 from https://www.pricesystems.com/about-us/

[62] The International Federation of Accountants. 2009. *Evaluating and Improving Costing in Organizations*. International Federation of Accountants (IFAC). Retrieved from https://www.ifac.org/knowledge-gateway/preparing-future-ready-professionals/publications/evaluating-and-improving-costing-organizations

[63] United Nations. 2014. *United Nations Practical Manual on Transfer Pricing for Developing Countries*. United Nations. Retrieved from https://www.un-ilibrary.org/content/books/9789210561372

[64] Muhammad Usman, Emilia Mendes, Francila Weidt, and Ricardo Britto. 2014. Effort estimation in agile software development. In *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*. ACM, 82–91. DOI: https://doi.org/10.1145/2639490.2639503

[65] J. M. Verner and N. Cerpa. 2005. Australian software development: What software project management practices lead to success? In *Proceedings of the 2005 Australian Software Engineering Conference*. IEEE, 70–77. DOI: https://doi.org/10.1109/ASWEC.2005.14

[66] Anna Wiedemann, Manuel Wiesche, and Helmut Krcmar. 2019. Integrating development and operations in cross-functional teams—toward a devops competency model. In *Proceedings of the 2019 on Computers and People Research Conference (SIGMIS-CPR'19)*. Association for Computing Machinery, New York, NY, 14–19. DOI: https://doi.org/10.1145/3322385.3322400

[67] Hong Wu, Lin Shi, Celia Chen, Qing Wang, and Barry Boehm. 2016. Maintenance effort estimation for open source software: A systematic literature review. In *Proceedings of the 2016 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 32–43. DOI: https://doi.org/10.1109/ICSME.2016.87

[68] Darja Šmite, Nils Brede Moe, Aivars Šāblis, and Claes Wohlin. 2017. Software teams and their knowledge networks in large-scale software development. *Information and Software Technology* 86 (2017), 71–86. DOI: https://doi.org/10.1016/j.infsof.2017.01.003

Received 25 July 2022; revised 25 April 2023; accepted 11 July 2023

B

A Research Model for the Economic Assessment of Inner Source Software Development

In the following, the full text of the article *A Research Model for the Economic Assessment of Inner Source Software Development* can be found. The article was originally published as:

[13] Stefan Buchner and Dirk Riehle. 2023. A Research Model for the Economic Assessment of Inner Source Software Development. In Proceedings of the 56th Hawaii International Conference on System Sciences, HICSS '23. ScholarSpace, Maui, Hawaii, 353–364. https://hdl.handle.net/10125/102672

A Research Model for the Economic Assessment of Inner Source Software Development

Stefan Buchner
Computer Science Department
Friedrich-Alexander Universität
Erlangen-Nürnberg, Germany
stefan.buchner@fau.de

Dirk Riehle
Computer Science Department
Friedrich-Alexander Universität
Erlangen-Nürnberg, Germany
dirk@riehle.org

Abstract

Inner source is the use of open-source practices within companies. It enables more efficient software development, shortens time-to-market, and lowers costs through increased company-internal collaboration. While existing studies examine social and organizational impact factors on inner source adoption, only a few have looked at measuring and economically assessing inner source. This article presents an overview of current research regarding inner source, its measurement, economic assessment, and impact on businesses and their processes. Based on a systematic literature review we build a research model for economic inner source assessment. This research model shows thematic dependencies between the economic impact of inner source and its measurement. Additionally, it proposes research questions and hypotheses on measuring, economically assessing, and subsequently adopting inner source.

Keywords: Inner source, open source, economic assessment, systematic literature review, research model

1. Introduction

When companies make use of inner source for their software development, they apply open-source practices within their organization (Capraro & Riehle, 2016). This means they open their repositories for internal re-use and incentivize developers of other teams and organizations to contribute to their software (Gruetter et al., 2018). In inner source, companies do not develop publicly available repositories as common in open source, but adopt its peer-review characteristics and early feedback cycles (Edison et al., 2020).

Inner source brings various advantages not only for development but also for organizational and general business aspects of companies. One important reason for adopting inner source is the higher code quality (Stol & Fitzgerald, 2015). It also increases employee satisfaction (Capraro, 2020; Riehle et al., 2016) and makes overall development, especially of software platforms (Riehle et al., 2016), more efficient. As a result, overall development time can be shortened, and costs reduced (Capraro & Riehle, 2016; Edison et al., 2020). Even though inner source is getting more popular recently (Edison et al., 2020), it is not widespread yet. The reason lies in both the development and business-operation side of organizations. On the one side, social and cultural challenges of inner source adoption are widely researched, but metrics are not (Edison et al., 2020). On the other side, inner source contributions are made at a high frequency, making it hard for businesses to adapt their processes accordingly (which our review also showed).

Previous research already made first attempts at measuring inner source collaboration (Capraro et al., 2018) and quantifying it for economic purposes (Buchner & Riehle, 2022). Nevertheless, only some work in the measurement domains exists, as Edison et al. (2020) found in their literature review. They called for more research on inner source metrics and real-life validation. In our research (in contrast to existing work), we look deeper into the topic of inner source metrics, especially related to economic challenges. The focus of this research is on the economic assessment of inner source, which can be defined as the quantification of inner source development work and its artifacts for economic business purposes (e.g. planning and operation). As this is done inadequately in current research, we asked the following research questions:

RQ1: What is the current state of research in economic inner source assessment?

RQ2: What are current challenges of economically assessing inner source and how can they be tackled?

This research briefly presents the economic impact of inner source and its measurement. Understanding this also helps mitigate risks that arise when inner source is

URI: https://hdl.handle.net/10125/102672
978-0-9981331-6-4

not properly applied and measured.

One of the most important risks of using inner source is being accused of profit shifting (Buchner & Riehle, 2022). Moreover, various important business processes and basic organizational principles are affected by the cross-boundary collaboration pattern of inner source and need to be adopted, as our research shows. Therefore, our paper provides not only an overview of the current research situation, but can also be important for avoiding greater risks in software engineering and management through increased collaboration across organizational and international boundaries.

The remainder of the article is structured as follows: Section 2 discusses related work followed by Section 3 describing the methodologies. Section 4 shows the result of the literature review. Section 5 then explains the research model builds on top of the review. After that, the results are discussed in Section 6. In the end, possible future research topics are shown in Section 7, followed by a conclusion in Section 8.

2. Related work

On a basic level, inner source is already well defined by numerous researchers (e.g. Carroll et al. (2018), Cooper and Stol (2018), Morgan et al. (2011), and Stol et al. (2014), Stol et al. (2011)). The same is true regarding benefits, challenges, and industry perspectives (Capraro & Riehle, 2016; Froment & de Lohéac, 2021; Morgan et al., 2021; Stol et al., 2011).

During our literature review, we looked into the impacts of inner source on measurement-related processes and metrics within businesses. In related business domains, previous work also defined many commonly used methods relevant to or affected by inner source.

In taxation for example the OECD and UN already defined commonly used methods for calculating the value of the intellectual property (IP) flowing between tax boundaries (the so-called transfer price) and related challenges for digital businesses (OECD, 2015, 2017; United Nations, 2014). For inner source, the first algorithms were designed to calculate such a value based on code contributions (Buchner & Riehle, 2022).

Accounting is one topic affected by inner source which is in general already well defined in industry and research e.g. by The International Federation of Accountants (IFAC, 2009). Especially cost-related processes are already well defined for contexts outside of inner source (e.g. absorption costing (Aurora, 2013)). However, research already proposed accounting models for software platforms (Kornberger et al., 2017), but not for inner source.

Additionally, management-related metrics and software engineering processes are well defined. For example, various KPIs exist to measure software engineering progress (Cheng et al., 2009), but not specifically for inner source. Beyond general software management practices (Jones, 2004; Quinnan, 1980; Verner & Cerpa, 2005) risk management is widely researched (Ebert et al., 2008; Kwak & Stoddard, 2004; Roy, 2004).

While various topics of central importance for companies (taxation, management, accounting) are generally well defined, only a few have directly considered inner source (e.g. transfer pricing). Our work lays the foundation for more work in those domains. Edison et al. (2020) already identified missing inner source metrics as future research topics. However, they only proposed general future research domains (e. g. inner source adoption, governance/management, methodologies, and practical application). In contrast to existing work, our literature review goes beyond showing the current state of inner source research in general but focuses on algorithms for effort estimation and prediction in businesses and how they are related to inner source businesses and their processes.

We saw in our first iteration of this work that measuring inner source is a topic placed between algorithmic implementation principles and economic impacts within businesses. In this paper, we not only briefly show the results of a systematic literature review that was conducted, but more importantly, we go beyond it. We mention additional implications and insights with a research model for economic inner source assessment. Our goal is to build a unified view based on the systematic literature review, connecting algorithmic and business perspectives on inner source measurement.

Consequently, our research:

- Provides a brief overview of the current research state of economic inner source assessment
- Connects the algorithmic/metric and business perspectives in inner source research
- Creates a research model showing the relations between the algorithmic and business perspectives
- Proposes future research based on the research model

3. Research method

For our research, we used the systematic literature review (SLR) approach of Kitchenham (2004) in combination with thematic analysis from Braun and Clarke (2006). Both emphasized an iterative/non-linear character, which we utilized by conducting three overall iterations.

3.1. Systematic literature review

Kitchenham (2004) divides the literature review process into several steps. It starts with several planning steps in advance of the review ((1) Identifying the need for a review, (2) specifying research questions, (3) developing and evaluating research protocol). Afterwards, the main research is conducted ((4)Identification of literature, (5) literature selection, (6) quality assessment, (7) data extraction and synthesis). At last, the results are reported (done here).

The research was conducted following the details of the research protocol typical for an SLR. Besides the already presented research questions and showing that a need to review exist, it contained the following aspects:

Databases: Our searches were conducted using Google Scholar, IEEE Xplorer, ACM Digital Library, Springer Link, Ebscohost, Wiley, and Scopus.

Identification process: We conducted three iterations. During our first iteration, we were able to identify that economic assessment in inner source is based on the two research domains business and algorithmic, which use different search terms and journals. The following iterations looked separately into each domain. This separation also shows in the results of the paper. Another important part of literature identification was forward and backward searches from previously found literature. Overall, the literature identification for inner source measurement and metrics is (especially due to the several research domains involved) more exploratory than common literature review based on a large data being narrowed down.

Quality criteria: Papers were included if they were peer-reviewed and in English. Additionally, contributions of recognized organizations (e.g. the OECD) were accepted. Papers were accepted if they meet the rigor and relevance criteria proposed by Ivarsson and Gorschek (2011).

Keywords: The used keywords were also different for business and algorithmic search domains. We saw early in our research that in the domain of measuring inner source almost no research exists leading to a large variety of keywords combinations. As our research is also not only touching one single domain, the insights of previous iterations helped to add more keywords for the next iterations. The following search terms were used to specify the development methods:

(Inner source OR open-source OR collaborative development OR cross-boundary collaboration OR cross border collaboration OR internal open-source OR software engineering OR software development OR DevOps OR agile OR platform)

These keywords were then combined with a variety of specific keywords for each search domain. For the business domain, the keywords are:

(business processes OR management OR accounting OR controlling OR taxation OR transfer pricing OR organization OR businesses OR enterprises OR organizational principles OR organization forms OR absorption costing OR cost calculation OR project management OR risk management OR product management)

For the algorithmic domain:

(Software development OR programming OR ((cost OR effort) AND (calculation OR prediction OR estimation OR measuring OR quantifying OR computing OR calculating)) OR measurement OR KPI)

Inclusion & exclusion criteria: We included papers which are either measurement-related inner source papers or business process papers affected through cross-boundary collaboration. Moreover, we included cost/effort calculations. We excluded papers that showed algorithms not being reproducible or applicable to the cross-boundary pattern of inner source. This mainly affects machine learning papers.

3.2. Thematic analysis

Thematic analysis by Braun and Clarke (2006) is a method for qualitative data analysis based on previously identified data sources (literature in our case). The goal is to identify common patterns in the data (called codes) and to classify them (called themes).

Braun & Clarke propose several ways on how the data analysis can be conducted. In our case, we chose a deductive approach as we come from a research question and look step by step closer at the business and economic topics. Additionally, we also made use of the iterative/non-linear pattern they described.

Overall, the thematic analysis consists of six steps: (1) getting familiar with the data, (2) generating initial codes, (3) creating candidate themes, (4) reviewing themes, (5) defining and naming themes, and (6) producing the report.

We conducted our research on two mainly independent coding processes, as the literature from the SLR showed the algorithmic and business differentiation, which also manifested in the created codes and themes.

3.3. Research method combination

We performed in our research a combination of SLR by Kitchenham and thematic analysis by Braun &

Clarke as they complement each other.

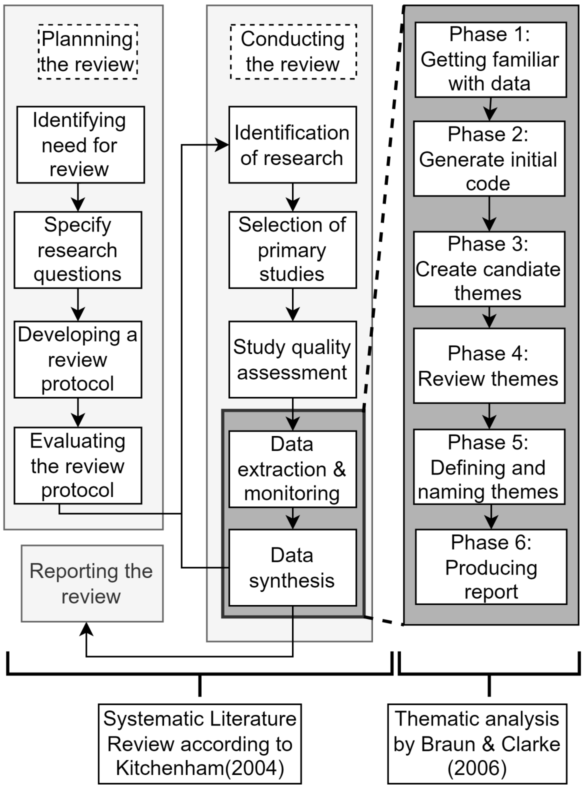

Figure 1. Overview of combined research process

Kitchenham sets a strong focus on identifying and selecting suitable literature but does not go into all detail about data analysis in their data extraction and synthesis steps. Braun & Clarke is solely focusing on data analysis. Therefore, we use thematic analysis embedded as data analysis within the systematic literature review. Figure 1 shows the research steps we performed and how the two used methods fit together. There we can also see the iterative character of the approach.

Combining the two methods also aligns well with the iterative approach we used. The result of each iteration was a network of themes and codes. Those were then used to identify missing aspects and literature. In the following iteration, especially literature in the missing domains was searched and included in the next coding phase. The overall process was performed until no new themes were found.

4. Systematic literature review results

Our review showed that economic assessment of inner source is embedded into the two domains of business and algorithms. In each domain, inner source relevant work gets published but in a largely isolated context. We found that only a small amount of published work applies to the high-frequency, cross-boundary collaboration pattern of inner source. Our combined SLR and thematic analysis process explored both domains independently from each other. However, during the data analysis process, important insights connecting the business and algorithmic domains showed up which will be explained later.

Overall, we looked at 49 papers. Figure 2 shows in which year the selected publications were published. We can see that many more recent papers were analyzed (especially inner source and algorithmic papers), but also some older papers set economic basics.

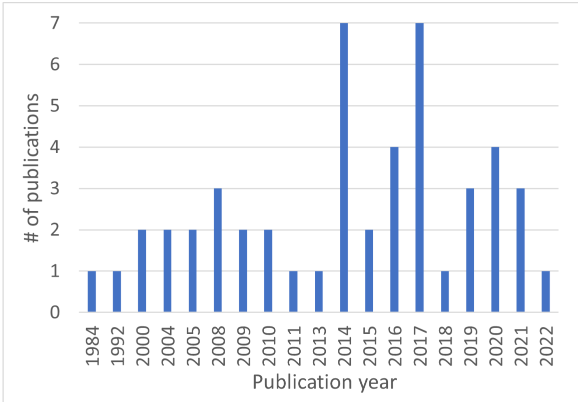

Figure 2. Number of publications per year

The codes and themes that emerged from the literature can be seen in Figure 3. Table 1 shows how many and which literature was used for which theme.

4.1. Business domain

From a business point of view, we found that inner source is embedded within various typical organization forms (matrix, functional, platform organization) affecting used software engineering methods (e.g. DevOps or agile development (Capraro, 2020; Wiedemann et al., 2019)). Additionally, inner source is extensively utilizing cross-boundary collaboration (Buchner & Riehle, 2022). We classified those three aspects (software development, cross-boundary collaboration, and organization forms) as theme *Business foundations*.

Besides the general embedding into the business, we were also able to identify three major domains which get affected when using inner source: Accounting (e.g. Astromskis et al. (2014) and Kornberger et al. (2017)), taxation (e.g. OECD (2017)), and management (e.g Jones (2004)). Those domains are not only a key part of successful business operations but also heavily

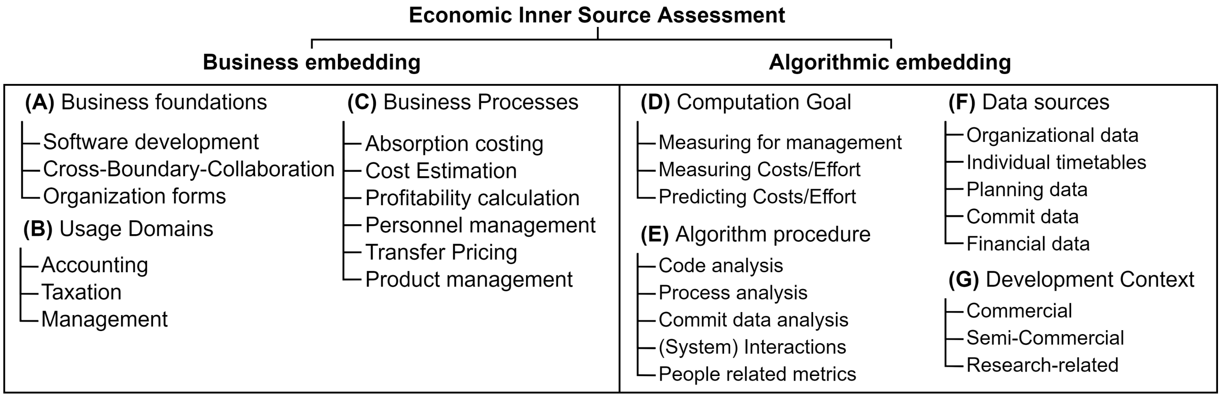

Figure 3. Themes and codes resulting from the thematic analysis in the form of a hierarchical code system

dependent on exact measurements of procedures taking place within the business. Theme *Usage domains* represents those domains in our review.

The reason why the domains of theme *Usage domains* are especially affected by inner source is due to the underlying daily business processes (Theme *Business processes*). We identified various example processes which are especially affected through the cross-boundary collaboration pattern of inner source. The already well-established models (see Section 2) can't be applied sufficiently anymore as cost calculation gets inaccurate with inner source. We identified operational processes (e.g. absorption costing (Aurora, 2013), profit calculations (IFAC, 2009), general cost estimations (Usman et al., 2014), transfer pricing (Buchner & Riehle, 2022)) and also more strategic business processes like personnel management (Riehle et al., 2016) and product management (Ebert, 2014).

4.2. Algorithmic domain

From an algorithmic perspective, our review looked at how applicable the reviewed algorithms were for inner source and related business challenges.

We identified different goals of individual algorithms (Theme *Computation goal*). While some algorithms are designed to measure management-related aspects (e.g. Basili et al. (2010) and Cheng et al. (2009)), some others do calculate costs or effort in general. Moreover, cost/effort-related calculations were often designed for either measuring historic data (e.g Gousios et al. (2008)) or making predictions (e.g. Karna et al. (2020).

Related to the computation goals we identified a variety of procedures to fulfill the different goals (Theme *Algorithm procedure*). While some algorithms analyze the written code itself (Astromskis et al., 2014), others look at the commit history (Buchner & Riehle, 2022; Moulla et al., 2021), related development and business processes (e.g. Dueñas et al. (2021) and Karna et al. (2020)), or system interactions (e.g. Wu et al. (2016)). Additionally, some papers identify or process metrics related to the developers themselves (Moulla et al., 2021; Qi et al., 2017).

The analyzed algorithms are using a variety of data sources (Theme *Data sources*). Most of them are well assessable and retrievable with a business context: commit data, financial data, organizational data, planning data, or individual timetables.

Lastly, we also found that algorithms were either created by commercial vendors (making it harder to reproduce e.g. PRICE Systems (2021)), semi-commercial (from commercial vendor published in a research paper e.g Boehm (1984)), or developed in research without industry background (Theme *Development context*).

4.3. Important findings

During the analysis, we showed that the domains related to inner source measurement (business and algorithmic) cannot be completely separated from each other. Even though literature originates from different research domains and shows different logical findings (codes), they also have some important underlying aspects in common.

One example is that investigated algorithms, specifically their development goals (Theme *Computation goal*) align with the processes affected by inner source (Theme *Business processes*). The business processes can also be classified as the goals most algorithms target. This shows us that both research domains are not only connected but depending on each other. The algorithms usable for cross-boundary collaboration are designed to fulfill specific business needs. On the other hand, businesses can only provide their services if they have algorithms available for their

Theme	# of Sources	Sources
Business foundations	17	Capraro (2020), Capraro and Riehle (2016), Edison et al. (2020), Feller and Fitzgerald (2000), Ford and Randolph (1992), Fuller (2019), Gruetter et al. (2018), Hobday (2000), Leite et al. (2020), Lindvall et al. (2004), OECD (2015), Riehle et al. (2016), Šmite et al. (2017), Stol et al. (2014), Stol et al. (2011), and Stol and Fitzgerald (2015), Wiedemann et al. (2019)
Usage domains	17	Astromskis et al. (2014), Aurora (2013), Buchner and Riehle (2022), Ceran et al. (2014), Ebert et al. (2008), Gruetter et al. (2018), Jones (2004), Kornberger et al. (2017), Kwak and Stoddard (2004), Mazur (2016), OECD (2015, 2017), Olbert and Spengel (2017), Plesner Rossing et al. (2017), IFAC (2009), United Nations (2014), and Verner and Cerpa (2005)
Business processes	18	Antolic (2008), Aurora (2013), Basili et al. (2010), Bilgaiyan et al. (2017), Buchner and Riehle (2022), Capraro (2020), Cheng et al. (2009), Ebert (2014), Karna et al. (2020), Neumann (2019), OECD (2015, 2017), Riehle et al. (2016), Stol et al. (2014), IFAC (2009), United Nations (2014), Usman et al. (2014), and Wu et al. (2016)
Computation goal	12	Antolic (2008), Astromskis et al. (2014), Basili et al. (2010), Bilgaiyan et al. (2017), Boehm (1984), Cheng et al. (2009), Gousios et al. (2008), Karna et al. (2020), Qi et al. (2017), Robles et al. (2014), Usman et al. (2014), and Wu et al. (2016)
Algorithm procedure	15	Antolic (2008), Astromskis et al. (2014), Basili et al. (2010), Bilgaiyan et al. (2017), Buchner and Riehle (2022), Cheng et al. (2009), Dueñas et al. (2021), Gousios et al. (2008), Kang et al. (2010), Karna et al. (2020), Moulla et al. (2021), Qi et al. (2017), Robles et al. (2014), Usman et al. (2014), and Wu et al. (2016)
Data sources	19	Antolic (2008), Astromskis et al. (2014), Basili et al. (2010), Buchner and Riehle (2022), Cheng et al. (2009), Dueñas et al. (2021), Kang et al. (2010), Karna et al. (2020), and Moulla et al. (2021)
Development context	3	Bilgaiyan et al. (2017), Boehm (1984), and PRICE Systems (2021)

Table 1. Overview of sources per theme

assessment. This is especially important for measuring and introducing inner source in companies, as the current state is not sufficient yet.

Moreover, the historic and predictive differentiation of algorithms (in theme *Computation goal*) is also visible on the business side. Cost estimation and management-related calculations are rather predictive while cost calculations are historic.

Another important finding is that even though we selected only algorithms generally applicable to inner source (Theme *Algorithm procedure*), only a few of them can easily be applied to it. Best suitable are algorithms directly capable of measuring cross-boundary collaboration (e.g. using commit data). However, numerous algorithms might give supportive information or need to be adapted to apply to inner source (e.g. analyzing process data is not adapted to cross-boundary collaboration in software engineering).

Overall we can see that less work on combining measurement and its economic impact on businesses and their processes exist for the inner source domain.

5. Inner source research model

5.1. Basics

In this section, we are showing an inner source research model built on the insights of the SLR. We found that algorithms are suitable for assessing inner-source-related operational and strategic processes. Here we channel back the insights to research by building an inner source research model for economic assessment topics. The goal is to build a unified understanding of challenges coming with measuring inner source development.

Based on the insights of the literature review, we created a research model. Palvia et al. (2006) identified several types of research models: Descriptive models, which are minimum models listing variables, and more complex prescriptive models with (hierarchical) relationships. We created a prescriptive model, an influence diagram in particular. We followed the formalization of Petter et al. (2007) for creating theoretical constructs and hypotheses. Those constructs are the basic aspects that define the research model, connected by hypotheses that need to be proven. In our case, constructs are theories in inner source and economic assessment which base on the codes on themes of the thematic analysis. The hypotheses show how the constructs influence each other.

The research model can be seen in Figure 4. Our model was developed in accordance with the themes and codes of our SLR. The structures and mentioned implicit connections there (e.g. Themes *Business processes* & *Computation goal*) also manifest in the research model: The differentiation between the algorithmic perspective (left side of the research model) and the business perspective (right side of the research model). Additionally, the frequently recognized difference between historic and predictive algorithms as well as the strategic and operational business processes can also be seen in the model.

The research model illustrates how the different perspectives (algorithmic/business, historic/predictive, strategic/operational) belong together. It consists of four basic hierarchies (from left to right): The first one is the algorithms on the left, being the basis for a larger inner source economic assessment model (middle hierarchy), which influences the strategic/operational business processes (right hierarchy). The fourth hierarchy (furthest to the right) shows a potential influence of our measuring inner source on inner source adoption. The hypotheses are those that need to be investigated in future research. In the following, we will explain those and their origin in more detail.

5.2. Research model

Algorithmic view: Our literature review shows different algorithm types for different use-cases/goals (Themes *Computation goal* & *Algorithm procedure*). We found out that only a few types of algorithms are suitable for assessing inner source. Mainly those, who make it possible to assess individual cross-boundary code flows (e.g. Code *Commit data analysis*) can be used for economic inner source assessment. However, algorithms not designed to handle cross-boundary collaboration (see especially Code *People related metric*) may only be used partly for inner source assessment. The thematic analysis revealed the connection of algorithmic procedures to a variety of business processes (Theme *Business processes*) manifesting as hypotheses that we will explain now.

While performing the thematic analysis, we were able to identify that examined algorithms looked into retrospectively-oriented and predictive purposes. Theme *Computation Goal* implicitly already introduced that timely differentiation. Hypotheses H1 and H2 build on that differentiation.

H1: The ability to use development-, system and process data to ***measure*** *software development correlates positively with the ability to economically assess IP transfer between organizational units*

The first important basis we identified were algorithms for measuring (retrospectively) historic software development characteristics, whereas these measurements might also include management-related metrics of historical performance (e.g. KPIs, GQM).

In this research model we propose based on our insights of the SLR that measuring (historic) software development and its processes (specifically adapted to the cross-boundary collaboration pattern) is an important part of the wider-spread economic assessment of IP transfer between organizational boundaries.

H2: The ability to use development-, system- and process data to ***predict*** *software development correlates positively with the ability to economically assess IP transfer between organizational units*

Complementary to H1, predictive procedures for inner source are also important for economic inner source assessment. The ability to predict software development work in various ways for different (previously explained) purposes solves problems which cannot be realized by only measuring historic data.

Business view: The algorithmic view differentiates between measuring historic and predicting future costs based on the insights of theme *Computation goal*. That differentiation also shows on the business side at theme *Business processes* and its codes. A comprehensive economic measurement model for software development can contain more than measuring costs e.g. various management metrics or inner source-specific metrics. An extensive evaluation of possible metrics still has to be made there.

We saw that measuring inner source is important for operational and strategic decisions (Theme *Usage domains*). Therefore, measuring inner source is not only important for operational business processes but also for long-term business success. The operational and strategic differentiation shown with the codes of the thematic analysis (especially theme *Business processes*) is the basis of Hypothesis H3 and H4 utilizing the algorithmic insights of H1 and H2.

H3: The ability to economically assess IP transfer between organizational units correlates positively with the usability of economically assessed inner source development for ***strategic*** *business purposes*

From the strategic perspective, we were able to identify that the lack of metrics and the middle management's fear of losing their performance goals (Riehle et al., 2016) is a driving force for assessing inner source. With management being important for adopting inner source, solving strategic inner source challenges can help thriving its adoption. Economically assessing inner source might lower its adoption boundaries by easing financial-related problems (Themes *Usage domains* & *Business processes*). Moreover, providing a better overview of historic development activities and making more precise predictions also enables easier inner source adoption.

Various challenges coming from inner source in

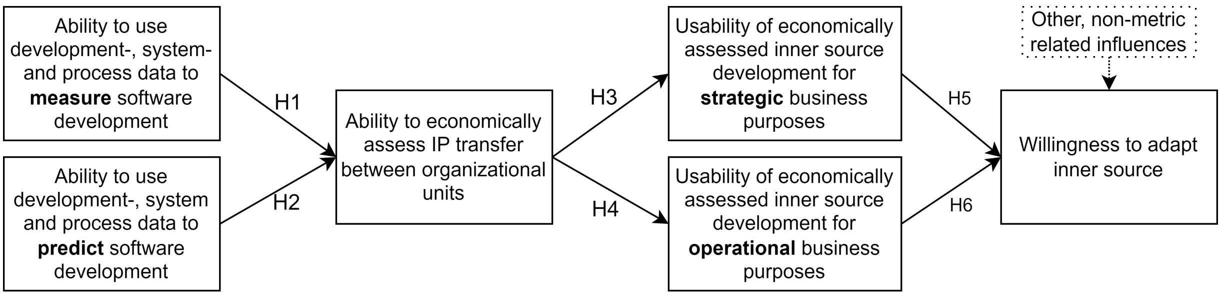

Figure 4. Inner source research model

businesses (see Theme *Business processes*) fit the strategic perspective covered by our model. The challenges originate from the domains of product management (estimating product-related aspects), personnel management (calculating workflow and performance of teams), and strategic management (better insights into achieved goals).

H4: The ability to economically assess IP transfer between organizational units correlates positively with the usability of economically assessed inner source development for ***operational*** *business purposes*

From an operational perspective, economically assessing inner source might help to keep established processes (Themes *Usage domains* and *Business processes*) applicable to the inner source paradigm. We showed with our SLR that inner source influences business processes within the whole organization, especially those related to cost calculation. Moreover, we explained how algorithms help to financially assess inner source development.

Inner source adoption: We saw a rising number of challenges within various domains (particularly accounting and taxation), which impact the inner source adoption rate due to long-term uncertainties such as the fear of unintended profit shifting (Buchner & Riehle, 2022). Consequently, overcoming corresponding organizational and strategic challenges may improve inner source adoption rates. Hypotheses H5 and H6 show that in the research model.

H5: The usability of economical assessed inner source development for ***strategic*** *business purposes correlates positively with the willingness to adapt inner source*

Edison et al. (2020) identified various influencing factors for inner source adoption by reviewing multiple studies. These factors include various domains outside of inner source measurement (e.g. knowledge management, cultural and management-related aspects). The research model might provide additional tools and insights from a strategic business perspective making inner source adoption more efficient.

H6: The usability of economical assessed inner source development for ***operational*** *business purposes correlates positively with the willingness to adapt inner source*

Similar to the strategic business perspective the ability to measure inner source for operational business purposes also might increase the willingness to adapt inner source.

6. Discussion

6.1. Implications

We asked two research questions for this paper. RQ1 (current state on economic inner source assessment) was answered through the SLR (Section 4). RQ2 (challenges in inner source assessment and how to tackle them) can be answered by looking into the details of the proposed research model.

Generally, our research model gives an overview of thematic connections between the business and algorithmic topics affected by inner source and identified through our SLR (Hypotheses H1 to H4). It shows how challenges coming from inner source (identified through the SLR) can be systematically tackled by looking into the proposed hypotheses.

For industry, the research model shows that algorithms used in business, their goals, and their design are closely related to how businesses are organized. They affect software development as well as how strategic and operational business decisions are conducted. This is important for inner source as it is deeply integrated into the company's organizational structure and software development. Therefore, inner source not only affects development but also strategic and operational decisions based on the yet-to-determine measurement and prediction algorithms.

Our research model also shows that measuring inner source is important beyond directly measurable strategic/operational processes. It sets the basis for more general software engineering measurement. Holistic inner source measurement might be an integral part of improving inner source adoption.

For research, the model not only provides an overview of potential future research (Section 7), but answers/discusses some open topics of past research. Edison et al. (2020) already defined inner source metrics as a field of interest for research. The research model builds on top of that providing additional insights including various business and economic details. Our research model provides additional value by deeply looking into the dependencies of algorithms in effort estimation and their connection to inner source business, which was not done in past research.

Measuring inner source is also an important factor for overcoming some of the identified inner source adoption challenges (e.g. middle managers' fear of losing their performance goals Riehle et al. (2016)) providing basic tools for all mentioned process-related challenges that lower inner source adaption rate.

Proving the hypotheses of the research model and developing an inner source measurement tool can provide a basic tool for future research. All kinds of software engineering-related hypotheses can be built on top of an inner source measurement model, even outside the inner source scope.

6.2. Limitations

We mainly looked at inner source measurement from a research point of view, as the goal was to develop a research model. Our research did not include primary data from the industry. Industry input was indirectly included through research papers working with industry. This allowed us to consider the industry perspective in an already evaluated way without losing focus on our literature review and analysis. We propose future work to explicitly consider industry perspectives through case studies or interviews.

Additionally, our research model took only literature related to the economic measurement of inner source into relation. Other influence factors, especially social factors impacting inner source and its measurement have not been considered. We specifically chose our research scope like that to keep focus during literature selection and analysis.

7. Future research

The research model also provides an overview of domains of particular interest for future research. The proposed areas of future research follow the logical outline of the research model and therefore are structured in the same way as Figure 4. By utilizing the proposed research model, future researchers have a plan at hand regarding which areas they might want to examine (algorithms in inner source, their integration into the businesses, and measuring and improving inner source adoption). Future research can then prove the hypothesis e.g. by conducting case studies where inner source is measured (measuring code repositories and other development artifacts), economically assessed (e.g. calculating costs and benefits of inner source), and integrated into improved business processes.

For the algorithmic side of inner source measurement (based on H1/H2 in the research model), we saw a need to develop algorithms better suitable for cross-boundary collaboration in general or inner source software engineering in specific. We also saw that future research needs to build new artifacts on top of those algorithms.

Additionally, future research needs to integrate developed inner source algorithms and tools into existing development processes for daily usage.

For the economic side of inner source measurement (based on H3/H4 in the research model), future research needs to identify strategic and operational requirements for a holistic strategic and operational inner source measurement model.

Moreover, future research needs to develop either new tools or adapt existing ones for various inner source affected domains and processes (accounting, tax, management, based on themes *Usage domains & Business processes*) to meet the identified requirements. Important is also to connect future inner source management tools and business process tools to those used for software engineering to provide a unified view of inner source and its effects.

Furthermore, we propose to use inner source measurement tools to investigate the effects of inner source on business performances. Comparing the results to traditional development might also be of interest to not only confirm increased development efficiency through inner source but also to identify domains where inner source still can be improved.

In the domain of inner source adoption (see Hypotheses H5 and H6), we propose future research to look into how inner source measurement affects and can improve inner source adoption. On top of that, we suggest researching methods and guides that practitioners can use to improve inner source within their businesses.

8. Conclusions

In this paper, we discussed a structured literature review on the topic of inner source, especially its measurement and economic assessment. We looked into the business process and algorithmic areas to get a comprehensive overview of related topics.

Of the literature review we conducted a thematic analysis to classify concepts and identify important relationships between those topics (codes and themes). We found that outside of software development inner source affects businesses in various aspects. We recognized that inner source mostly influences cost-related processes, mainly within accounting, management, and tax. However, even though many algorithms for measuring and predicting effort exist, only a few of them are suitable for the application within inner source. Existing algorithms need to be adapted to apply to inner source. Based on those measurements, more tools and algorithms need to be developed.

To bring the open topics in the algorithmic and business side of inner source measurement together we build a research model for economic inner source assessment. It shows the connection and relationship between algorithmic measurement, its impact from a business perspective as well as inner source adoption in the long place.

Moreover, we gave a brief overview of possible new research topics based on our research model. Therefore, our research sets the basics for better measurement of software engineering and understanding its implications for future research and industry.

9. Acknowledgement

This work was supported by the DFGs (German Research Foundation) Research Grants Programme (Management Accounting for Inner Source, RI 2147/5-2).

We thank our colleagues for continuous feedback and proofreading of this article.

References

Antolic, Z. (2008). An example of using key performance indicators for software development process efficiency evaluation. *R&D Center Ericsson Nikola Tesla*, *6*, 1–6.

Astromskis, S., Janes, A., Sillitti, A., & Succi, G. (2014). An approach to non-invasive cost accounting. *2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications*, 30–37.

Aurora, B. B. C. (2013). The cost of production under direct costing and absorption costing – a comparative approach. *Annals - Economy Series*, *2*, 123–129.

Basili, V., Lindvall, M., Regardie, M., Seaman, C., Heidrich, J., Münch, J., Rombach, D., & Trendowicz, A. (2010). Linking software development and business strategy through measurement. *Computer*, *43*, 57–65.

Bilgaiyan, S., Sagnika, S., Mishra, S., & Das, M. (2017). A systematic review on software cost estimation in agile software development. *JOURNAL OF ENGINEERING SCIENCE AND TECHNOLOGY REVIEW*, *10*, 51–64.

Boehm, B. W. (1984). Software engineering economics. *IEEE Transactions on Software Engineering*, *SE-10*(1), 4–21.

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology*, *3*, 77–101.

Buchner, S., & Riehle, D. (2022). Calculating the Costs of Inner Source Collaboration by Computing the Time Worked. *Proceedings of the 55th Hawaii International Conference on System Sciences, HICSS '22*.

Capraro, M. (2020). *Measuring inner source collaboration* (Doctoral dissertation). Friedrich-Alexander-Universität Erlangen-Nürnberg.

Capraro, M., Dorner, M., & Riehle, D. (2018). The patch-flow method for measuring inner source collaboration. *Proceedings of the 15th International Conference on Mining Software Repositories*, 515–525.

Capraro, M., & Riehle, D. (2016). Inner source definition, benefits, and challenges. *ACM Comput. Surv.*, *49*(4).

Carroll, N., Morgan, L., & Conboy, K. (2018). Examining the impact of adopting inner source software practices. *Proceedings of the 14th International Symposium on Open Collaboration*.

Ceran, Y., Dawande, M., Liu, D., & Mookerjee, V. (2014). Optimal software reuse in incremental software development: A transfer pricing approach. *Management Science*, *60*(3), 541–559.

Cheng, T.-H., Jansen, S., & Remmers, M. (2009). Controlling and monitoring agile software development in three dutch product software companies. *2009 ICSE Workshop on Software Development Governance*, 29–35.

Cooper, D., & Stol, K.-J. (2018). *Adopting innersource: Principles and case studies*. O'Reilly Media.

Dueñas, S., Cosentino, V., Gonzalez-Barahona, J. M., del Castillo San Felix, A., Izquierdo-Cortazar, D., Cañas-Diaz, L., &

Garcia-Plaza, A. P. (2021). GrimoireLab: A toolset for software development analytics. *PeerJ Computer Science*, *7*.

Ebert, C. (2014). Software product management. *IEEE Software*, *31*(3), 21–24.

Ebert, C., Murthy, B. K., & Jha, N. N. (2008). Managing risks in global software engineering: Principles and practices. *2008 IEEE International Conference on Global Software Engineering*, 131–140.

Edison, H., Carroll, N., Morgan, L., & Conboy, K. (2020). Inner source software development: Current thinking and an agenda for future research. *Journal of Systems and Software*, *163*, 110520.

Feller, J., & Fitzgerald, B. (2000). A framework analysis of the open source software development paradigm. *Proceedings of the Twenty First International Conference on Information Systems*, 58–69.

Ford, R. C., & Randolph, W. A. (1992). Cross-functional structures: A review and integration of matrix organization and project management. *Journal of Management*, *18*(2), 267–294.

Froment, T., & de Lohéac, G. A. (2021). The convergence of struggles! reusability assessment of inner-source components for product lines. *INSIGHT*, *24*(1), 30–34.

Fuller, R. (2019). Functional organization of software groups considered harmful. *2019 IEEE/ACM International Conference on Software and System Processes (ICSSP)*, 120–124.

Gousios, G., Kalliamvakou, E., & Spinellis, D. (2008). Measuring developer contribution from software repository data. *Proceedings of the 2008 International Working Conference on Mining Software Repositories*, 129–132.

Gruetter, G., Fregonese, D., & Zink, J. (2018). Living in a biosphere at robert bosch. In D. Cooper & K.-J. Stol (Eds.), *Adopting innersource: Principles and case studies*. O'Reilly Media.

Hobday, M. (2000). The project-based organisation: An ideal form for managing complex products and systems? *Research Policy*, *29*(7), 871–893.

Ivarsson, M., & Gorschek, T. (2011). A method for evaluating rigor and industrial relevance of technology evaluations. *Empirical Software Engineering*, *16*(3), 365–395.

Jones, C. (2004). Software project management practices: Failure versus success. *CrossTalk: The Journal of Defense Software Engineering*, *17*(10), 5–9.

Kang, S., Choi, O., & Baik, J. (2010). Model-based dynamic cost estimation and tracking method for agile software development. *2010 IEEE/ACIS 9th International Conference on Computer and Information Science*, 743–748.

Karna, H., Gotovac, S., & Vicković, L. (2020). Data mining approach to effort modeling on agile software projects. *Informatica*, *44*(2).

Kitchenham, B. (2004). Procedures for performing systematic reviews. *Keele, UK, Keele University*, *33*(2004), 1–26.

Kornberger, M., Pflueger, D., & Mouritsen, J. (2017). Evaluative infrastructures: Accounting for platform organization. *Accounting, Organizations and Society*, *60*, 79–95.

Kwak, Y., & Stoddard, J. (2004). Project risk management: Lessons learned from software development environment. *Technovation*, *24*(11), 915–920.

Leite, L., Kon, F., Pinto, G., & Meirelles, P. (2020). Platform teams: An organizational structure for continuous delivery. *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, 505–511.

Lindvall, M., Muthig, D., Dagnino, A., Wallin, C., Stupperich, M., Kiefer, D., May, J., & Kahkonen, T. (2004). Agile software development in large organizations. *Computer*, *37*(12), 26–34.

Mazur, O. (2016). Transfer pricing challenges in the cloud. *Boston College Law Review*, *57*(2), 643–693.

Morgan, L., Feller, J., & Finnegan, P. (2011). Exploring inner source as a form of intraorganisational open innovation. In V. K. Tuunainen, M. Rossi, & J. Nandhakumar (Eds.), *19th european conference on information systems, ECIS 2011, helsinki, finland, june 9-11, 2011* (p. 151).

Morgan, L., Gleasure, R., Baiyere, A., & Dang, H. P. (2021). Share and share alike: How inner source can help create new digital platforms. *California Management Review*, *64*(1), 90–112.

Moulla, D. K., Abran, A., & Kolyang. (2021). Duration estimation models for open source software projects. *International Journal of Information Technology and Computer Science*, *13*(1), 1–17.

Neumann, A. (2019). *Transfer pricing in inner source software development* (Master's thesis). Hochschule des Bundes für öffentliche Verwaltung. Bruhl, Germany.

OECD. (2015). *Aligning transfer pricing outcomes with value creation, actions 8-10 - 2015 final reports*. OECD. https://www.oecd-ilibrary.org/content/publication/9789264241244-en

OECD. (2017). *Oecd transfer pricing guidelines for multinational enterprises and tax administrations 2017*. OECD. https://www.oecd-ilibrary.org/content/publication/tpg-2017-en

Olbert, M., & Spengel, C. (2017). International taxation in the digital economy : Challenge accepted? *World Tax Journal : WTJ*, *9*(1), 3–46.

Palvia, P., Midha, V., & Pinjani, P. (2006). Research models in information systems. *Communications of the Association for Information Systems*, *17*(1).

Petter, S., Straub, D., & Rai, A. (2007). Specifying formative constructs in information systems research. *MIS Quarterly*, *31*(4), 623–656.

Plesner Rossing, C., Cools, M., & Rohde, C. (2017). International transfer pricing in multinational enterprises. *Journal of Accounting Education*, *39*(100), 55–67.

PRICE Systems. (2021). Company overview: Price® systems [Retrieved: 2021-12-15]. https://www.pricesystems.com/about-us/

Qi, F., Jing, X.-Y., Zhu, X., Xie, X., Xu, B., & Ying, S. (2017). Software effort estimation based on open source projects: Case study of github. *Information and Software Technology*, *92*, 145–157.

Quinnan, R. E. (1980). The management of software engineering: Part v: Software engineering management practices. *IBM Syst. J.*, *19*(4), 466–477.

Riehle, D., Capraro, M., Kips, D., & Horn, L. (2016). Inner source in platform-based product engineering. *IEEE Transactions on Software Engineering*, *42*(12), 1162–1177.

Robles, G., González-Barahona, J. M., Cervigón, C., Capiluppi, A., & Izquierdo-Cortázar, D. (2014). Estimating development effort in free/open source software projects by mining software repositories: A case study of openstack. *Proceedings of the 11th Working Conference on Mining Software Repositories*, 222–231.

Roy, G. G. (2004). A risk management framework for software engineering practice. *2004 Australian Software Engineering Conference. Proceedings.*, 60–67.

Šmite, D., Moe, N. B., Šāblis, A., & Wohlin, C. (2017). Software teams and their knowledge networks in large-scale software development. *Information and Software Technology*, *86*, 71–86.

Stol, K.-J., Avgeriou, P., Babar, M. A., Lucas, Y., & Fitzgerald, B. (2014). Key factors for adopting inner source. *ACM Trans. Softw. Eng. Methodol.*, *23*(2).

Stol, K.-J., Babar, M. A., Avgeriou, P., & Fitzgerald, B. (2011). A comparative study of challenges in integrating open source software and inner source software. *Information and Software Technology*, *53*(12), 1319–1336.

Stol, K.-J., & Fitzgerald, B. (2015). Inner source–adopting open source development practices in organizations: A tutorial. *IEEE Software*, *32*(4), 60–67.

The International Federation of Accountants. (2009). *Evaluating and improving costing in organizations*. International Federation of Accountants (IFAC). https://www.ifac.org/knowledge-gateway/preparing-future-ready-professionals/publications/evaluating-and-improving-costing-organizations

United Nations. (2014). *United nations practical manual on transfer pricing for developing countries*. United Nations. https://www.un-ilibrary.org/content/books/9789210561372

Usman, M., Mendes, E., Weidt, F., & Britto, R. (2014). Effort estimation in agile software development. *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*, 82–91.

Verner, J., & Cerpa, N. (2005). Australian software development: What software project management practices lead to success? *2005 Australian Software Engineering Conference*, 70–77.

Wiedemann, A., Wiesche, M., & Krcmar, H. (2019). Integrating development and operations in cross-functional teams - toward a devops competency model. *Proceedings of the 2019 on Computers and People Research Conference*, 14–19.

Wu, H., Shi, L., Chen, C., Wang, Q., & Boehm, B. (2016). Maintenance effort estimation for open source software: A systematic literature review. *2016 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, 32–43.

C

Calculating the Costs of Inner Source Collaboration by Computing the Time Worked

In the following, the full text of the article *Calculating the Costs of Inner Source Collaboration by Computing the Time Worked* can be found. The article was originally published as:

[12] Stefan Buchner and Dirk Riehle. 2022. Calculating the Costs of Inner Source Collaboration by Computing the Time Worked. In Proceedings of the 55th Hawaii International Conference on System Sciences, HICSS '22., virtual, 7466–7475. https://doi.org/10.24251/HICSS.2022.896

Proceedings of the 55th Hawaii International Conference on System Sciences | 2022

Calculating the Costs of Inner Source Collaboration by Computing the Time Worked

Stefan Buchner
Computer Science Department
Friedrich-Alexander University
Erlangen-Nürnberg, Germany
stefan.buchner@fau.de

Dirk Riehle
Computer Science Department
Friedrich-Alexander University
Erlangen-Nürnberg, Germany
dirk@riehle.org

Abstract

A key part of taxation, controlling, and management of international collaborative programming workflows is determining the costs of a supplied software artifact. The OECD suggests the use of the Cost Plus method for calculating these costs. However, in the past, this method has been implemented using only coarse-grain data from the costs of whole organizational units. Due to the move to inner source software development, we need a much more fine-grain solution for computing the detailed time spent on programming specific components. This is necessary, because a more accurate work time distribution is required to fulfill the fiscal and administrative challenges posed by collaborating across organizational boundaries. In this article, we present a novel method to determine the time spent on an individual code contribution (commit) to a software component for use within cost calculation, especially for taxation purposes. We demonstrate the usefulness of our approach by application to a real-world data set gathered at a large multi-national corporation. We evaluate our work through feedback received from this corporation and from the German Ministry of Finance.

1. Introduction

In software developing companies, engineering managers, the finance department, and human resources all would like to know how long programming tasks took in the past and may take in the future. The reasons are manifold: An engineering manager may need this information to develop a project plan, the finance department may need this information to calculate the cost of a software component, and the human resources department may want to know this information to determine performance.

Today's common solution is to let software developers self-declare by way of checking-in and checking-out of certain work tasks, or simply by filling in time-sheets. The introduction of inner source software development by software vendors as a software development approach that complements and extends existing practices has made this insufficient.

Inner source software development is the use of open source best practices inside companies. No open source software is being developed, only its practices are being used. For this, departments open up their code base to the whole company, typically on an internal forge like GitHub, advertise their components, welcome visitors, and engage with them in the hope that such visitors will find these components useful. The goal is to get visitors to start using a component so that they will eventually contribute to it, which leads to cost reduction to both the original developers and the visitor-turned-user-turned-collaborator [1]. Other demonstrated benefits of such inner source collaboration are developing higher quality software components within the company, better knowledge sharing, and higher employee satisfaction, among others [2][3][4].

In this new world of inner source, developers not only work on tasks that have been assigned to them by their managers, they also work on and contribute to software components across the organization, often crossing organizational boundaries, even tax boundaries. The number of components a developer contributes to can increase significantly. In this situation, it becomes impossible for a developer to precisely track how they spend their time for programming different components. The number of possible components they contribute to is too large for all practical purposes. It would be better if the calculation of time spent on programming a particular code contribution could be automated.

Being able to measuring time spend on commits and assigning an economic value (e.g. costs) to it helps companies mastering their financial challenges posed by new cross-boundary software development. A solution which is able to calculate the costs of inner source might also be helpful for various accounting and profit calculations (where costs play a central role, see [5]) as well as management related challenges (e.g.

URI: https://hdl.handle.net/10125/80238
978-0-9981331-5-7

product management [4] [6], risk management [7] or KPI calculation [8]). Prospectively, economic inner source assessment can help companies to manage and introduce new organization forms, business processes, information systems and developing their business overall.

In this article, we present a method to compute the time spent on programming a particular code contribution (commit to a code repository) for usage within cost calculation. It is robust towards the developer switching gears and contributing to multiple different components in short sequence.

With inner source being so closly related to economic assessment and business processes, we asked the following research question:

RQ: How can we calculate the time spend on code contributions for usage within various cost related business processes

We take a design science approach and motivate our work through the use case of calculating transfer prices for inner source contributions in globally distributed software development for a client/supplier relationship.

The remainder of this article is structured as follows: In section 2 we present related work, in section 3 we present our research design, in section 4 we identify the problem and define a research objective, in section 5 we present a solution design and its implementation, and in section 6 we first demonstrate and then evaluate our solution. In section 7 we discuss the research limitations and in section 8 we present our final conclusions.

2. Related Work

Our algorithm connects the topics inner source software development and transfer pricing. Therefore, we review research on these topics as related work.

2.1. Transfer pricing

Even though our solution might be applicable to large variety of cost calculation purposes in management and controlling, the main motivator when designing the algorithm was to calculate transfer prices, especially for taxation purposes.

While in an ordinary market, two market participants determine the price of a product or (intellectual) property in orientation to the overall market, for transferring property within one organisation, no such market exists. However, as those transactions are often performed between tax boundaries, standards for calculating the so called transfer price exist to ensure fair pricing and taxation according to the function and risk of each transfer [9] [10].

As an international standard, the OECD defined five standard methods for calculation of transfer prices [9]: The Comparable Uncontrolled Price Method, Resale Price Method, Cost Plus method as Traditional Transaction Methods or Transactional Net Margin method and Transactional Profit Split Method as Transactional Profit Methods. The exact method used is determined by the function, risk and overall situation a transaction is situated in [11].

With the growing importance of software and digital business in the economy, problems in transfer pricing are arising, which do not only affect the calculation of prices themselves, but also the broader economic influence on earnings of countries [12][13][14]. Therefore, the algorithm provided in this paper contributes both to cost calculation for taxation (and other business related cost and management problems) and to solve more general taxation questions.

The algorithm presented in this paper calculates work times for usage in cost calculation. From the five standard transfer pricing methods we chose Cost Plus for our implementation and demonstration as it is based on cost calculation [9]. Cost Plus is a method, where the transfer price is determined by calculating all the costs occurring in producing the transferred good or property. One commonly used way to calculate costs is the full absorption costing, which differentiates all costs of a business unit between costs directly assignable to a product or service (in our case direct to the transaction) or those who cannot be directly assigned to a product (indirect costs) [15]. For Cost Plus, the sum of direct and partial indirect cost is calculated, before a profit margin is added on top. Our approach helps to split the indirect costs of a business unit according to the real work effort for each product, as the problem statement will show more in detail (Section 4.1). Therefore, our solution targets calculating the work distribution between certain projects and not the time spent on each individual commit.

2.2. Inner source software development

Inner source is the use of open source best practices inside a company [2].

Like in open source, in inner source, developers lay open all project or product artifacts for everyone to see (only within the company, not publicly like in open source) [16][17]. They want other developers to find their code (open source typical self-selection) and start using it. The hope is that users identify bugs and help the code mature. Eventually, code contributions might flow back to the original developers (through peer-reviews), helping share development costs [3] [1].

As a literature review from Edison et al. [18] showed, inner source and its research is deeply integrated within a wide range of business aspects including knowledge & business management, business model design, and collaboration measurement. Being able to determine costs and software related intellectual property flow therefore is not only a matter of traditional, already well defined costing, accounting and management methods (e.g. [19] [20] [5] [8]), but is also important for inner source within industry [21].

Use of, contribution to, and collaboration on inner source projects is not apriori restricted. Thus it can happen across organizational boundaries, as developers begin to look past their silo. Such silo boundaries might be the boundaries of legal entities and hence any collaboration across these boundaries might be taxation relevant and hence require the calculation (and payment) of transfer prices.

3. Research Approach

We take a design science approach for our research. Design science is a methodological research framework that is used when researchers not only want to empirically analyze a situation, but also want to develop innovative solutions to real-world problems. Of several defined design science methodologies, we choose Peffers et al. [22], because of its ease of use.

Design science, according to Peffers et al. [22] is an iterative process, consisting of six main steps (plus an additional communication step, omitted here). The process centers on the creation and evaluation of an innovative artifact. This design science artifact should be a novel solution for an identified problem. The steps correspond with the following activities:

1. Problem identification. The researcher identifies and motivates a problem. In section 4, we present our past work and current literature review to identify the need for determining the (financial) costs of inner source collaboration.

2. Objective definition. The researcher defines the objectives for a solution to the identified problem. Our objectives (presented in section 4) are building on the problem identification and therefore are originating from the needs identified with our partners.

3. Solution design. The researcher develops a novel and innovative design science artifact according to the objectives. The solution we developed is based on statistical data gathered with our industry partner. The result is a method including an algorithm, presented in section 5.

4. Implementation. The researcher implements the solution for purposes of demonstration and evaluation. Implementation in our case indeed is the software implementation of the method and the algorithm, also presented in section 5.

5. Demonstration. The researcher first demonstrates the solution by applying the artifact to at least one instantiated problem. From past work, we use a real-world data set to demonstrate our solution, and present it in section 6.

6. Evaluation. The researcher evaluates the artifact using an appropriate method. We use member checking to gather feedback on our novel method and present the results in section 6.

The design science process allows for iteration, and this is also how we performed our work. In our case, we statistically analyzed gathered data, implemented prototypes and doing interim evaluations for ourselves in iterations. The presentation in the following sections is mostly linear, however, and focuses on the results.

4. Problem and Objective

The first and second step of the design science approach are the problem identification and objective definition, which will be explained in detail in the following two section.

4.1. Problem Identification

The problem identification was done in cooperation with our partners who later evaluated our approach with us. Additional literature research finalizes the problem identification. We saw, that inner source development can be taxation relevant (See 2.2). As inner source collaboration happens between business units, a need for calculating costs of transferred work between these units exist.

In the past it was sufficient to calculate costs of business entities coarse-grain, as no fine-grain differentiation between different projects of one business unit was needed. This changes with inner source, as contributions to software owned by different organizational and business units might be made. Moreover, the contributions are by the minute and not only once in retrospect for a larger period of time. Additionally, inner source contributions cannot be determined in advance (in cost and size). Three interviews conducted with industry partners through the data gathering process in previous work confirmed that these problems do occur in daily business. One interviewee was a software architect, one interviewee was a (Scrum) product owner, and one other interviewee was an engineering manager. The interviews showed that calculating transfer prices is up to now solved by roughly estimating the costs rather than determining them more exactly through an algorithm. This leads to insecurities in terms of fiscal correctness and management acceptance.

In addition to calculating transfer prices, our solution might contribute to improving team management, utilizing inner source advantages (e.g. [23] [24] [1]) and handle organizational challenges coming with software development: For instance, functional organization is considered harmful for software engineering [25] and platform-based teams are still bringing many implementation barriers with it, as middle managers fear to not reach personal performance goals, if contributions to outside units are made [4]. Future research must show, how this paper can best be used to help introducing inner source within companies.

Finally, measuring working time for commits helps splitting indirect costs (e.g. personnel costs of developers) for usage within full absorption costing (see section 2.1) more exactly, as such a measurement is only coarse-grain up to now (see sections 1 and 2).

4.2. Objective Definition

Our literature review and industry interviews, as presented in the problem identification section, showed an unresolved mismatch between the need for correctly pricing software supply relationships and the demands of inner source software development and high-frequency code contributions across taxation boundaries. To that end, we define the objective for our research:

To develop a new implementation of the Cost Plus method that

- can correctly determine transfer prices for code contributions in client-supplier relationship, and
- can handle high frequency code collaboration between client and suppliers,

where "high frequency" means many times daily as it is common in inner source software development collaboration. As we have the goal to develop an algorithm for cost calculation we want to find a way how to measure the distribution of the working time per project rather than measuring the exact time spent on each commit.

5. Design and Implementation

The third design science step is to present the solution design (section 5.1). After that (fourth step, section 5.2) the implementation will be explained.

5.1. Solution Design

Developing the solution design was done in several iterations by repeatedly analyzing commit data gathered in cooperation with our industry partners and improving the algorithm proposed in the previous iteration. The idea of the solution in this paper is to calculate working times from commits, which then might be used in different use cases. As this paper focuses on the use case of calculating transfer prices (using Cost Plus) for inner source development, the main result to be calculated is the cost share. Therefore, the basic solution design targets more an exact cost share for each organizational unit, then exact working times per commit. This also means, that organizational overhead will not be excluded in the calculation as the assumption is, that the overhead for each project is distributed equally to the work effort put into the project.

The concept was developed using commit data of a large multi-national corporation containing commits to a software platform from about 400 developers organized in 94 organizational units spreading over four hierarchy levels. The data-set was gathered continuously over one and a half years containing 230,000 file changes of 29,000 commits from 13 inner source projects/components. Due to the large amount of available data we assume, that the commit behavior (e. g. commit times, intervals between commits, LOC per commit) is representative for development work happening in companies. In future research we want to verify data and concept by conducting further studies in other multinational companies.

Required data. The following information are needed to design and implement our algorithm:

- Commit data (e.g. from Git): Author, Lines of Code (added, modified, deleted), timestamp, file path, commit identifier, project identifier
- Organizational hierarchy, incl. headcount
- Project list incl. owning organization
- Developer list and their organizational unit

Development process. We developed the concept by iteratively performing statistical analysis of the commit data and interim evaluations of implemented prototypes.

After all, we were able to identify two points of view: 1) Looking at the time difference between two commits of the same author and 2) Looking at the commit timestamps. As first analysis we plotted the time differences of two commits from the same author. The results show, that most of the commits were made frequently within minutes or hours (mostly within 12 hours/720 minutes), but also significant number of commits are made once a day (culminating at about 1440 minutes/24hours). The same pattern can be observed for larger time differences (mostly commits with up to seven days in between). Moreover we found out (through analyzing the Lines of code committed and number of files changes) that committing at least once a day is the usual commit behaviour and that commits longer apart were e.g. mostly weekends or holidays. For the timestamp analysis, the commits were

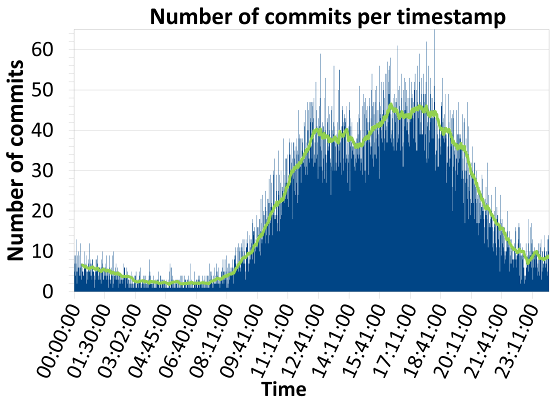

Figure 1. Number of commits per timestamp

grouped and counted by its commit time. Figure 1 shows the plot. Most commits are made during the day, but the sample data also contain commits made during the night. The data show, that developers are having flexible working times, which must be considered during further calculations.

Additionally performed analysis steps during will be described at the appropriate chapter.

Concept design. This chapter is about how working times are calculated. Cost calculation is done in the implementation step. The working time calculation concept is based on the time difference and timestamp distributions. Our previous results showed, that most commits are made at least daily. Resulting from that insight, the commits are mainly differentiated by time difference into two groups.

The first group are those, which are one working day (36 hours/2160 minutes, resulting from the previous analysis) or less in time difference. This time difference also includes commits being done in the evening with the previous commit being in the morning of the previous day. In short, all commits that belong to this group are those that regularly contribute to the software on a daily basis.

The second group of commits are all those, which are irregular. This is either the case if the time difference to the previous commit is longer than 36 hours apart or if the commit has no time difference at all (single commits by one author). Consequently, this group for example is suited for all first commits after weekends, holidays or long time without contribution, before beginning frequent development again.

Regular working time assignment. All regular commits (time difference <2160 minutes) are differentiated again by the time difference, as our previous analysis results have shown, that different use-cases are given, depending on the commits timestamp and the time difference. Additionally, it is more likely that commits with a small time difference are having a longer working times than those covering a night with larger time differences. In detail, three cases are differentiated:

1. <360 min. time difference
2. >=360 min. & <= 720 min. time difference
3. >720 min. & <= 2160 min. time difference

<360 minutes: The first group of commits (26% of all commits) are those with time difference less than 6 hours. These commits (mostly made within a matter of minutes) are getting working time equally to their time difference assigned, as there is a close timely relationship between the previous and the current commit.

>720 minutes & <= 2160 minutes time difference: The second group (mentioned third above, 22.7% of the commits) are all commits being 12 to 36 hours after the previous one. These commits are most likely to cover at least one night as they span a greater range of time as our analysis have confirmed. Consequently, these commit cannot have working time assigned in height of their time difference.

Commits belonging to this group are calculating working time proportional to the typical commit time distribution. Basis for the calculation is are the number of commits per timestamp (See Figure 1). The idea is to calculate the number of commits made betweeen the timestamps of the current and previous commit. The number of commits are summed up and set in proportion to the overall number of commits recognized. This ratio is then set into proportion to the length of a typical workday. Putting the algorithm into an equation, results in:

$$wt_{prop} = \frac{\sum_{i=timestamp_{cur}}^{timestamp_{prev}} \mathrm{n}_i}{\mathrm{n}_{overall}} * \mathrm{wt}_{day} \qquad (1)$$

wt represents the working time (of a working day day/ the result $prop$), calculated by iterating over the timestamp of the current (cur) until previous ($prev$) commit, with being n the number of commits.

The effect of applying the working time proportional to the number of commits is, that commits having a time difference of exact 24 hours, get a whole typical working day assigned. If the time difference is larger (e.g. 1 1/2 working days), the commit gets more working time. Additionally, this formula also minds, that some developers prefer working into the night by calculating the night time as working time as it is typical over all commits.

>=360 minutes & <= 720 minutes: The third group (mentioned secondly above, 4.7% of the commits) are all commits between 6 and 12 hours after the previous commit. This means, that there is not necessarily a night between these commits with developers either having a long work day or having a shorter night. The data set showed, that commits between both variants are not unusual and hence must be differentiated.

Commits belonging to the 6 to 12 hours group are treated differently depending on the share of the night they are covering. For this calculation, at first, it must be calculated how many minutes of the night are covered by the commit($nightshare = ns$):

$$ns = \frac{\sum_{i=timestamp_{cur}}^{timestamp_{prev}} x_i * 1}{\sum_{i=timestamp_{nightBegin}}^{timestamp_{nightEnd}} 1}$$
$$x_i = \begin{cases} 1, & \text{if } i \in nighttime, \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

This formula counts for a commit, how many of the total minutes of a night (denominator) were covered (numerator). The result (a percentage) is used to calculate the working time:

$$wt_{ns} = ns * wt_{prop} + (1 - ns) * daycoverage$$
$$daycoverage = \sum_{i=timestamp_{cur}}^{timestamp_{prev}} x_i \tag{3}$$
$$x_i = \begin{cases} 1, & \text{if } i \in daytime, \\ 0, & \text{otherwise} \end{cases}$$

The logic behind calculating with nightshare is, that a commit gets as much proportional working time (see Formula 1) as it covers the night (ns). This ensures, that commits covering large parts/the whole night get working time typical for the night assigned. The minutes belonging to the day ($daycoverage$, sum of minutes during the day), are fully assigned.

Example: The night is 10pm - 7am. Commit 1 is at 8am, the previous one at 9pm. As it covers the full night ($ns = 1$), it is fully proportional calculated. Commit 2 is at 4pm, the previous one at 6am. Commit 2 covers only 11% of the night ($ns = 0.11$), consequently this amount is calculated proportional. The minutes during the day (7 am to 4 pm) are calculated to 89%.
Comparing these two examples, we can see, that Commit 1 (done early in the morning) got less working time. Commit 2 (rarely goes into the night) got almost regular working time assigned.

For the calculation, we need to set two timestamps as the beginning and end of the night. For the implementation, this might be done by setting those as a fix value (depending on the dataset/ business entity) or by calculating a meaningful start of the day automatically. The latter was chosen in our implementation by calculating it dynamically through looking at the number of commits per timestamp (Figure 1) and setting the night begin/end to a certain percentage (15% in our case) of the highest number of commits.

Irregular working time assignment. All commits without time difference (1.4% of the cases) or not daily committed (45% of the cases) need a different handling as they don't have any (close) timely relationship to other commits, on which the calculation can be made. As part of the earlier described development process, additional analysis where done in later iterations to dive deeper into the commit behavior of developers.

Building on the previously described development steps, the relationship of working times, time differences and LOC of all regular commits were plotted. Figure 2 shows, that the number of LOC is exponentially rising, but the time differences and working times are linear. Consequently, we can conclude, that it is not possible, to assign working time just by looking at the LOC of a commit. For every LOC, a wide range of time differences & working times are possible, and for every time difference or working time, a wide range of LOC are possible. Thereafter, average and quartile working times for each LOC where plotted (Figure 3). This gives a deeper look into the working time distribution and LOC relationship. Through calculating linear regression lines, the overall trend is visible more easily, resulting into close median and average working times per LOC. Assuming, that the data representative, the median regression line can now be taken to estimate the working time for all irregular commits. Using this method, the irregular commits also get working time

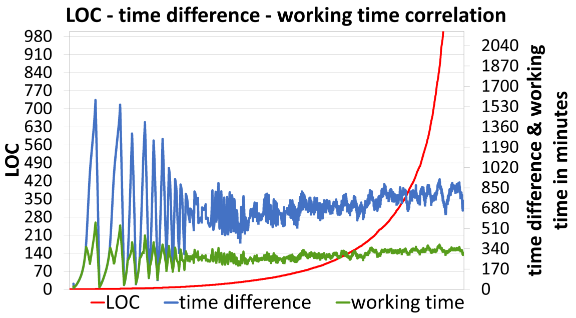

Figure 2. LOC, time difference and working time correlation

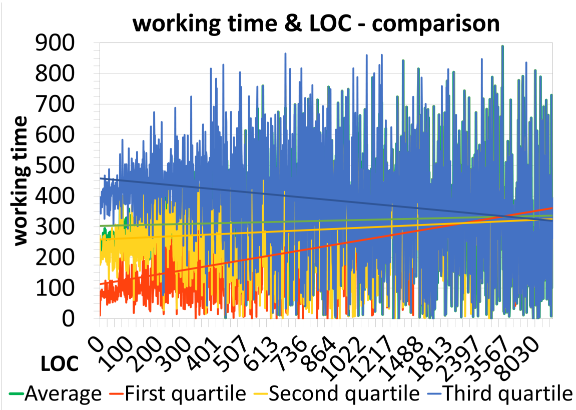

Figure 3. LOC and working time comparison

assigned based on the results of the regular commits:

$$wt_{linear} = 0.04267525 * loc + 258.58249058 \quad (4)$$

5.2. Implementation

For the implementation, the commit data (available in CSV format) where parsed and uploaded to a PostgreSQL database alongside all other information needed (See section 5.1). To make the data easier reusable in the future, a REST API was developed in PHP, returning the transfer price in JSON format, running on an Apache Webserver. The steps not only represent the processing order, but also the development process of the implementation.

The **first step** in the process was to upload the CSV commit data into the database. In this step, preprocessing was done to eliminate invalid data.

The **second step** was to receive cost data for each business entity (cost centre view). In our sample implementation no real-life data but dummy costs were used to show the basic cost calculation procedure.

The **third step** was to load the commit data from the database, process them (e.g. calculate time difference) and calculate the algorithm presented in the solution design section (Section 5.1). The result after this step is a list of commits, on which every commit has its working time assigned according to our solution design.

As already said, for our sample use case, share of the workload between all contributed projects of entity is important. Therefore, the **fourth step** was to aggregate the list of working times to the needed organizational level. We aggregated our sample data on a team-level, as this hierarchy level is the lowest available cost centre level. For each entity, the sum of working times per projects are calculated. Out of the sum of working times, the work share distribution can be calculated (e.g. 30% to Project A, 60% to Project B, 10% to Project C).

The **fifth step** was the calculation of the transfer prices, in our example using the Cost Plus method. This means, that for each entity, a value must be calculated which flows to other entities. The result of step four (work share distribution) is used for the calculation. As we also have the costs per business entity, we can now split all costs which are not directly assignable to one transaction (indirect costs e.g. developers salary) according to the work share spend on it. On top of all the costs, a profit margin is added (5% flat in our case as an demonstration example).

The end result is a list of organizational units and their costs per project they are developing. These data are returned in the **sixth step** over the REST API in JSON format and may be used in various use cases like controlling or taxation.

6. Demonstration and Evaluation

The last two steps in design science are the solution demonstration and evaluation.

6.1. Demonstration

We demonstrate our solution using the real world data on which the algorithm is based on. Before the cost calculation itself will be demonstrated, the work time assignment for each of the four cases (3 regular commits, 1 irregular without time difference) will be shown. The first example shows part of a JSON output from the API belonging to **Case 1** (time difference <360 min.):


```
{"module": "Module A",
"commit_date": "2015-01-02 15:18:02",
"owner": "Person A",
"time_difference": "23",
"loc": "462",
"working_time": "23"}
```


As the commit was done close after the previous commit, it gets its time difference as work time assigned. The implemented API returns not only the working time of the commit, but also additional data (e.g. time difference, name of committer, the module and LOC). Additionally, organizational data for the later cost calculation are included (not shown here).

Case 2 (360 - 720 min. time difference) can be shown with a commit, which was done in the morning after covering large parts of the night (previous commit was at 10:40 pm). Consequently, the commit does get less work time assigned for the night time. In our demonstration we can see, that the algorithm works exactly like desired: Commits covering the night get not the full night as work time:


```
{"module": "Module B",
"commit_date": "2016-02-15 09:35:27",
"owner": "Person B",
"time_difference": "655",
"loc": "70",
"night_share": 1,
"working_time": 50}
```


In this example, the commit covers 655 minutes, mostly over night. Due to the calculations of our algorithm, it gets 50 minutes of work time assigned, which means the developer effectively started programming (according to the statistical correlation to other commits) at 8:45 AM.

Case 3 (720-2160 min. time difference) is shown on a commit covering part of two work days (covering 16 hours through the night). Therefore, work time proportional to the statistical typical commits covering that time is assigned:


```
{"module": "Module C",
"commit_date": "2016-04-08 11:07:01",
"owner": "Person C",
"time_difference": "953",
"loc": "684",
"working_time": 154}
```


In this example, the previous commit was done at 7:14 PM and therefore covers not only the night, but also parts of the current work day. Our algorithm assigns the commit working time of 154 minutes, which (if the developer stopped working at 7:14 PM) effectively means the committer started working on the module at 8:33 AM.

Case 4 (Irregular commit) shows, that commits with no or to large time difference get reasonable work time based on the other commits by using the median regression line from the Solution section:


```
{"module": "Module D",
"commit_date": "2015-01-06 17:58:19",
"owner": "Person D",
"time_difference": null,
"loc": "227",
"working_time": 268}
```


In this case, the commit to module D was made by Person D, which only made a single commit to the software platform. The commit got work time (268 minutes) assigned in height of the median value of all regular commits (demonstrated with case 1 to 3) with equal LOC.

All working times are aggregated for each cost centre. In our second demonstration, the lowest organizational level with dedicated cost centres was taken. All commits belonging to this organization (committer is located there) are aggregated according to the target organization (owner of the module). Based on the total work times per organization, a working time share is calculated, which represents the amount of work every module the organization worked on. The percentage share is then used for splitting the indirect costs in that respective ration. After adding the profit margin (5% flat in our case), the transfer price can be calculated:


```
{"org_name": "Organisation-A-A-A",
"parent_org": "Organisation-A-A",
"transfer_data": {
"2015": {
"Organisation-A-A": {
"share": 84.96,
"costs": 47701.28,
"transfer_price": 50086.34
},
"Organisation-A-A-A": {
"share": 7.31,
"costs": 4106.04,
"transfer_price": 4311.34
},
"Organisation-B-A": {
"share": 6.53,
"costs": 3669.06,
"transfer_price": 3852.51
}[...]}}}}
```


This example transfer prices for the rather small sample entity show that in 2015 most of the commits (92.27%) are still within the same organization (*Organisation-A-A-A*) or for the parent business unit (*Organisation-A-A*). A smaller part (in sum 7.73%) is crossing tax boundaries to other business units (*B-A, etc.*) and therefore might be taxation and accounting relevant.

Commits to projects, which were labelled by the company as not relevant to the analysis where excluded in the analysis and consequently also not included in the transfer price calculation.

6.2. Evaluation

We evaluated our approach using member checking for our demonstration data and feedback from experts on the subject matter. In total, we conducted interviews with six people having three different views on transfer pricing in inner source: Two people form the German Ministry of Finance, one person from a large international account firm and three people from our industry partner (previously involved in problem identification). Our goal was to evaluate two main aspects: 1) Evaluated the basic usage of Cost Plus for transfer pricing within inner source. 2) Evaluate the algorithmic suitability of the algorithm itself.

Throughout our interviews and further collaboration with the German Ministry of Finance, we were able to analyze the different possible transfer pricing methods and its prerequisites to be met when applied to inner source development. A result is that Cost Plus is applicable (and first choice) for inner source development, when a client-supplier relationship is given [26]. Our interview with the large international accounting firm confirmed that, even though they emphasized, that choosing a method strongly depends on the individual case. We consider our evaluation objectives to be fulfilled, as we specifically targeted Cost Plus.

Our second main evaluation goal (suitability of the algorithm for calculating cost plus) was also evaluated with our interview partners at the German Ministry of Finance. The head of education for the department confirmed that our solution is not only a valid way of calculating Cost Plus for transfer pricing, but also offers a significant advantage compared to the way of determining the Cost Plus transfer price up to now. This confirmed the usability of our algorithm from a tax officials point of view.

Evaluating the algorithmic suitability from an industry point of view was done again with the three industry interview partners already involed in data gathering and problem identification (member checking, details to interviewed persons see section 4.1). As original data source they knew the data, its structure, and its corporate context. They confirmed that our results fit the real world. Additionally, they attest to us that the working time share was useful and fits the internal work flows. They challenged the algorithms ability to correctly calculate the time spent on an individual commit. However, in the context of calculating the Cost Plus method, we only care about the working time distribution (percentage-shares) and not individual commits. Hence, their general confirmation of the usefulness of our approach stands.

7. Discussion and Limitations

We present a novel algorithm and its implementation for more precise Cost Plus calculation. The method is defined by the OECD and is commonly used in calculating transfer prices in client supplier relationships. We move past current practice, which takes a coarse-grain approach, by basing our algorithm on the actual code contributions. Using our algorithm, we can calculate the time and associated labor costs to determine transfer prices between a supplier and their client. The demonstration and evaluation show that we succeeded on both our defined objectives: A new and correct algorithm that performs well when faced with a large amount of individual (but high-frequency) code contributions.

Additionally, our demonstration and evaluation also scopes this research: We do not move into acceptance or analyzing wide-spread use of our work (yet). Hence, empirical evaluation criteria like trustworthiness for qualitative research [27] or reliability and validity for quantitative research do not apply. We therefore limit our discussion to general aspects of our work.

The most obvious limitation is that on the one hand, we aim to provide a new algorithm for precisely calculating labor costs in software development. Yet, towards this general goal, we are missing the final step, which is to determine the error of the time spent on an individual code contribution. This error is most likely a distribution around the value we calculate, but we have yet to determine this. For the specific purposes of our research, as set out in the objective section, this missing piece does not matter: For calculating Cost Plus precisely, the percentage relationships are sufficient so that we don't need the absolute values.

In terms of computational efficiency, the algorithm scales linearly with the number of transactions (commits). Given that it is our goal to be precise and fine-grain, we have to go down to the level of accessing each individual commit, and hence a linear performance is the best we can get.

8. Conclusions

As already shown in the discussion and limitations section, we developed a new algorithm to determine working times spend on commits for usage within cost calculation, especially Cost Plus in transfer pricing.

Even though, the algorithm and its implementation is limited to calculate work distribution for organizations, it sets the basis to enable exact work time calculation per commit. Therefore, future research can focus on improving the accuracy of individual commits, which enables more application fields for the algorithm.

Moreover, future research needs to show which additional areas can benefit from our algorithm, what adjustments need to be made. From an inner source perspective it is worth finding out, how the results of this paper can best be used to improve acceptance of inner source and bring economic advantages with it.

9. Acknowledgment

We would like to acknowledge Elçin Yenişen Yavuz, Thomas Wolter and the anonymous reviewers for their valuable feedback that helped us improve the paper significantly.

References

[1] M. Capraro and D. Riehle, "Inner source definition, benefits, and challenges," *ACM Comput. Surv.*, vol. 49, Dec. 2016.

[2] K.-J. Stol and B. Fitzgerald, "Inner source–adopting open source development practices in organizations: A tutorial," *IEEE Software*, vol. 32, no. 4, pp. 60–67, 2015.

[3] M. Capraro, *Measuring Inner Source Collaboration*. PhD thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), 2020.

[4] D. Riehle, M. Capraro, D. Kips, and L. Horn, "Inner source in platform-based product engineering," *IEEE Transactions on Software Engineering*, vol. 42, pp. 1162–1177, 12 2016.

[5] The International Federation of Accountants, *Evaluating and Improving Costing in Organizations*. 2009.

[6] C. Ebert, "Software product management," *IEEE Software*, vol. 31, no. 3, pp. 21–24, 2014.

[7] C. Ebert, B. K. Murthy, and N. N. Jha, "Managing risks in global software engineering: Principles and practices," in *2008 IEEE International Conference on Global Software Engineering*, pp. 131–140, 2008.

[8] T.-H. Cheng, S. Jansen, and M. Remmers, "Controlling and monitoring agile software development in three dutch product software companies," in *2009 ICSE Workshop on Software Development Governance*, pp. 29–35, 2009.

[9] OECD, *OECD Transfer Pricing Guidelines for Multinational Enterprises and Tax Administrations 2017*. 2017.

[10] United Nations, *United Nations Practical Manual on Transfer Pricing for Developing Countries*. United Nations, 2014.

[11] Y. Holtzman and P. Nagel, "An introduction to transfer pricing," *The Journal of Management Development*, vol. 33, 02 2014.

[12] OECD, *Aligning Transfer Pricing Outcomes with Value Creation, Actions 8-10 - 2015 Final Reports*. 2015.

[13] O. Mazur, "Transfer pricing challenges in the cloud," *Boston College Law Review*, vol. 57, no. 2, pp. 643–693, 2016.

[14] M. Olbert and C. Spengel, "International taxation in the digital economy : challenge accepted?," *World Tax Journal : WTJ*, vol. 9, no. 1, pp. 3–46, 2017.

[15] B. B. C. Aurora, "The Cost Of Production Under Direct Costing And Absorption Costing – A Comparative Approach," *Annals - Economy Series*, vol. 2, pp. 123–129, April 2013.

[16] Open Source Initiative, "The open source definition," 2007.

[17] K.-J. Stol, P. Avgeriou, M. A. Babar, Y. Lucas, and B. Fitzgerald, "Key factors for adopting inner source," *ACM Trans. Softw. Eng. Methodol.*, vol. 23, Apr. 2014.

[18] H. Edison, N. Carroll, L. Morgan, and K. Conboy, "Inner source software development: Current thinking and an agenda for future research," *Journal of Systems and Software*, vol. 163, p. 110520, 05 2020.

[19] B. B. C. Aurora, "The cost of production under direct costing and absorption costing – a comparative approach," *Annals - Economy Series*, vol. 2, pp. 123–129, 2013.

[20] M. Kornberger, D. Pflueger, and J. Mouritsen, "Evaluative infrastructures: Accounting for platform organization," *Accounting, Organizations and Society*, vol. 60, pp. 79–95, 2017.

[21] G. Gruetter, D. Fregonese, and J. Zink, "Living in a biosphere at robert bosch," in *Adopting InnerSource: Principles and Case Studies* (D. Cooper and K.-J. Stol, eds.), 06 2018.

[22] K. Peffers, T. Tuunanen, M. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of Management Information Systems*, vol. 24, pp. 45–77, 01 2007.

[23] N. Carroll, L. Morgan, and K. Conboy, "Examining the impact of adopting inner source software practices," in *Proceedings of the 14th International Symposium on Open Collaboration*, OpenSym '18, (New York, NY, USA), Association for Computing Machinery, 2018.

[24] D. Cooper and K.-J. Stol, *Adopting InnerSource: Principles and Case Studies*. 06 2018.

[25] R. Fuller, "Functional organization of software groups considered harmful," in *2019 IEEE/ACM International Conference on Software and System Processes (ICSSP)*, pp. 120–124, 2019.

[26] A. Neumann, "Transfer pricing in inner source software development," Master's thesis, Hochschule des Bundes für öffentliche Verwaltung, Bruhl, Germany, 2019.

[27] E. G. Guba, "Criteria for assessing the trustworthiness of naturalistic inquiries," *ECTJ*, vol. 29, p. 75, Jun 1981.