Managing Technical Debt Using Intelligent Techniques - A Systematic Mapping Study

Danyllo Albuquerque, Everton Guimarães, Graziela Tonin, Pilar Rodriguez, Mirko Perkusich, Hyggo Almeida, Angelo Perkusich and Ferdinandy Chagas

Abstract—Technical Debt (TD) is a metaphor reflecting technical compromises that can yield short-term benefits but might hurt the long-term health of a software system. With the increasing amount of data generated when performing software development activities, an emergent research field has gained attention: applying Intelligent Techniques to solve Software Engineering problems. Intelligent Techniques were used to explore data for knowledge discovery, reasoning, learning, planning, perception, or supporting decision-making. Although these techniques can be promising, there is no structured understanding related to their application to support Technical Debt Management (TDM) activities. Within this context, this study aims to investigate to what extent the literature has proposed and evaluated solutions based on Intelligent Techniques to support TDM activities. To this end, we performed a Systematic Mapping Study (SMS) to investigate to what extent the literature has proposed and evaluated solutions based on Intelligent Techniques to support TDM activities. In total, 150 primary studies were identified and analyzed, dated from 2012 to 2021. The results indicated a growing interest in applying Intelligent Techniques to support TDM activities, the most used: Machine Learning and Reasoning under uncertainty. Intelligent Techniques aimed to assist mainly TDM activities related to identification, measurement, and monitoring. Design TD, Code TD, and Architectural TD are the TD types in the spotlight. Most studies were categorized at automation levels 1 and 2, meaning that existing approaches still require substantial human intervention. Symbolists and Analogizers are levels of explanation presented by most Intelligent Techniques, implying that these solutions conclude a general truth after considering a sufficient number of particular cases. Moreover, we also cataloged the empirical research types, contributions, and validation strategies described in primary studies. Based on our findings, we argue that there is still room to improve the use of Intelligent Techniques to support TDM activities. The open issues that emerged from this study can represent future opportunities for practitioners and researchers.

Index Terms—Technical Debt, Intelligent Techniques, Technical Debt Management Activities, Systematic Mapping Study.

1 Introduction

Technical Debt (TD) refers to technical compromises that can yield short-term benefits but might hurt the long-term health of a software system [1]. TD can be associated with different software artifacts (e.g., requirement, source code, and test artifacts) and span over different development phases [2]. Managing its impact includes identifying, monitoring, and measuring TD symptoms [3].

TD is always present in real-world software development. Even though Technical Debt Management (TDM) is a critical activity [1], many organizations do not have established TDM practices. Project managers and developers are longing for methods and tools to help them strategically plan, track, and pay down TD occurrences [4]. Given the diversity of everyday practices in software development, TDM can be complex since it relies on a decision process based on multiple and heterogeneous data, which are hard to be gathered and synthesized.

Based on this scenario, appropriate techniques must retrieve and handle data from various software artifacts to assess whether TD has occurred or is beyond a defined threshold. Moreover, deciding when and how TD occurrences should be paid is based on several objectives such as Lead Time, Maintenance Costs, and Risk mitigation. In other words, software organizations need techniques that analyze, process and make decisions based on extensive data from multiple sources to support TDM activities.

According to the challenges mentioned above, there is a promising opportunity to use Intelligent Techniques to support TDM activities. For this study's purposes, we defined Intelligent Techniques as those that explore data for knowledge discovery, reasoning, learning, planning, natural language processing, perception, or supporting decision-making [5]. Examples of Intelligent Techniques are expert systems, case-based reasoning, neural networks, genetic algorithms, probabilistic methods, and fuzzy systems. It is essential to mention that there is no complete consensus on what makes a technique "intelligent". An advantage of using this term over more popular ones such as "Artificial Intelligence" (AI) is because it is more comprehensive. The term is also used as an "umbrella" to group a set of techniques, including AI-based techniques in general and other techniques out of this classification (e.g., Analytic Hierarchy Process and Dynamic Programming).

- *D. Albuquerque, M. Perkusich, H. Almeida, A. Perkusich and F. Chagas are with the Intelligent Software Engineering Group (ISE/VIRTUS), Federal University of Campina Grande, Campina Grande, Paraiba, Brazil.*
  E-mails: {danyllo.albuquerque, mirko, hyggo, perkusic, ferdinandy}@virtus.ufcg.edu.br
- *E. Guimarães is with The Pennsylvania State University, Malvern, Pennsylvania, USA.*
  E-mail: ezt157@psu.edu
- *G. Tonin is with federal University of Fronteira Sul, Chapecó, Santa Catarina, Brazil.*
  E-mail: grazzi@ime.usp.br
- *P. Rodriguez is with Polytechnic University of Madrid, Madrid, Spain.*
  E-mail: pilar.rodriguez@upm.es

Manuscript received October XX, 2022; revised October XX, 2022.

Although Intelligent Techniques have a great potential to optimize TDM activities, there is little empirical knowledge of how these techniques can benefit them. To address this gap, we conducted a Systematic Mapping Study (SMS) based on the guideline presented by Petersen et al. [6] to explore the intersection of both fields and pointing out new and future research directions. This article describes the protocol of the employed SMS and reports our findings focusing on:

— investigating how the existing solutions apply Intelligent Techniques to support TDM;

— exploring which Intelligent Techniques currently support TD types and TDM activities;

— understanding the risks involved in using Intelligent Techniques-based solutions to support TDM activities;

— gathering the contributions, empirical research type, and validation strategies of studies that report the application of Intelligent Techniques to support TDM activities;

— analyzing our result's implications for industry practitioners and researchers; and

— documenting current research challenges and gaps for future research.

Finally, this article is an extension of study originally presented at the International Conference on Technical Debt 2022 (TechDebt'22) [7]. The remainder of this article is organized as follows. Section 2 describes the main concepts required to understand this study. Section 3 discusses related work. Section 4 presents the methods employed to conduct this SMS. Section 5 points out the results and presents our main findings. Section 6 discusses the results from practitioners' and researchers' perspectives. Section 7 discusses the study's threats to validity and the actions we took to mitigate them. Finally, Section 8 presents our final remarks and future work.

2 Background

This section discusses the TD fundamentals, its types, and activities required for TD Management (Section 2.1). Next, we introduce the concept of Intelligent Techniques and explain the classification framework adopted in this study (Section 2.2).

2.1 Technical Debt

TD concept. The term was introduced by Cunningham [8] when discussing with stakeholders the consequences of releasing poorly written code snippets to accelerate the development process. For example, even though the source code meets the system requirements in the current release, the consequences of TD occurrences might spread over other software components in case of future changes. That is, TD occurrences can negatively affect maintainability and evolvability [1].

The TD metaphor is used to communicate with non-technical stakeholders (e.g., corporate managers and clients, among others) about immature software artifacts and their impact. TD can be associated with the decision-making process about the shortcuts and workarounds taken in software development. However, these decisions can positively or negatively influence the quality of the artifacts produced as the outcome of the software development. TD items can occur due to many factors, such as a lack of team members' knowledge of writing the source code without following a specific programming style. Therefore, software organizations must perceive and manage TD in their projects [9].

TD types. Li et al. [10] classified TD into ten coarse-grained types, and each of those was further classified into several subtypes based on the root causes of TD. Next, we describe the most prominent TD types according to the classification mentioned above:

— *Design TD* refers to technical shortcuts in detailed design.

— *Code TD* is poorly written code that violates best coding practices or coding rules. Examples include duplicated and complex code.

— *Architectural TD* is caused by architecture decisions that compromise internal quality aspects, such as maintainability.

— *Test TD* refers to shortcuts taken in testing. An example is the lack of tests (e.g., unit tests, integration tests, and acceptance tests).

— *Requirements TD* refers to the distance between the optimal requirements specification and how these requirements are satisfied based on the system implementation under domain assumptions and constraints.

Other TD types are mentioned in the original study [10], such as Build TD, Documentation TD, Infrastructure TD, Versioning TD, and Defect TD. These TD types will also be discussed in our SMS (Section 5), but here we focused on describing the most recurrent ones.

Technical Debt Management (TDM). It includes a set of activities that: (i) prevent potential TD from being incurred; (ii) deal with the accumulated TD to make it visible and controllable; and (iii) balance cost and value in repayment TD occurrences. Li et al. [10] classified TDM into eight coarse-grained activities, being the most prominent described as follows:

— *TD Identification* detects TD caused by intentional or unintentional technical decisions in a software system through specific techniques, such as static code analysis.

— *TD Measurement* quantifies the benefit and cost of known TD in a software system through estimation techniques or estimates the level of the overall TD in a system.

— *TD Monitoring* watches the changes in the cost and benefit of unresolved TD over time.

— *TD Repayment* resolves or mitigates TD in a software system by techniques such as re-engineering and refactoring.

— *TD Prioritization* ranks identified TD according to specific predefined rules to support deciding which TD items should be (re)paid first and which TD items can be tolerated until later releases.

Besides those TDM activities listed above, Li et al. [10] mention others such as TD Communication, TD Prevention, and TD Representation/Documentation. These TD types and TDM categorization served as a basis for our SMS to understand the application of Intelligent Techniques and to what extent they support different TD types.

2.2 Intelligent Techniques

Concept. The term comprises techniques used to explore data for knowledge discovery, reasoning, learning, planning, perception, or supporting decision-making [5]. These techniques tolerate imprecision, uncertainty, partial truth, and approximation [11]. The guiding principle of Intelligent Techniques is to exploit this tolerance to achieve tractability, robustness, and low computation cost [12]. Moreover, Intelligent Techniques mimic consciousness in many important cases: (i) they can learn from experience; (ii) they can be popularized in areas where there is no direct experience; and (iii) they speed up input/output mapping with sequential analysis views.

On the other hand, Conventional Techniques commonly use the principles of accuracy, certainty, and stiffness. These techniques are based on mathematical or analytical models, binary logic, transparent systems, numerical analysis, and transparent software [12]. All input data, products, and processes are clearly defined. However, using only these techniques to manage real problems is challenging since they process raw data without reasoning about the problem and do not consider the system's historical or contextual information [11]. Table 1 shows a comparative analysis highlighting the main characteristics of Conventional and Intelligent Techniques [12], [11].

Table 1
Conventional Techniques versus Intelligent Techniques.

Conventional Techniques	Intelligent Techniques
Require an analytical model to accurately describe traditional computations, which require a lot of processing time.	Allowance for irrationality, uncertainty, partial truth, and approximation.
Although the calculations are complex, imprecision and uncertainties are undesirable qualities.	Allowance for imprecision and uncertainty can provide management capabilities, low cost, intelligent computing quotas, and savings in communications.
It is deterministic, that is, there is an accurate forecast for the outputs obtained due to the inputs used in the program.	It is stochastic, that is, there is no forecast for the outputs obtained due to the inputs used in the program.
Require accurate input.	Process uncertainty data.
Provide the exact answer.	Provide approximate answers.

Classification. Machine Learning, Expert Systems, Genetic Algorithms, and Fuzzy Systems are typical examples of Intelligent Techniques since they simulate human behavior, including the ability to learn, accomplish tasks, and emulate human expertise and decision-making. A complete list of Intelligent Techniques and their concepts are described in Table 2. In general, these techniques may be used to:

— acquire the individual and collective knowledge and extend a knowledge base using artificial intelligence and database technologies;

— collect and analyze the tacit knowledge using expert systems, case-based reasoning, and fuzzy logic;

— enable knowledge discovery, or discovering underlying, hidden patterns in data sets, using neural networks and data mining;

— generate solutions to highly complex problems using genetic algorithms;

— automate routine tasks using intelligent agents.

Furthermore, it is important to mention that there is no complete consensus in the literature on a broader nomenclature encompassing various techniques considered to be "intelligent". We are using this nomenclature as it has been used in other recent studies [11], [5], [12] and appears to be the most recurrent (and suitable) nomenclature to enclose this category of techniques. For this study, we did not consider techniques not listed by such studies [11], [5], [12] as being intelligent (e.g., formal methods, software process simulation modeling, and model-driven development) since they do not adhere to the characteristics of Intelligent Techniques adopted in this study (See Table 1).

3 Related Work

This section discusses secondary studies that focused on Technical Debt Management (TDM) (Section 3.1) and investigated how Intelligent Techniques have been applied to solve TDM problems (Section 3.2). Finally, it summarizes how our study complements the existing body of knowledge (Section 3.3).

3.1 Technical Debt Management

Technical Debt and its management have been increasingly investigated in recent years due to the existence of several secondary studies in the area [20], [21], [10], [22], [3]. Next, we discuss these studies and underlying results chronologically.

Tom et al. [20] presented a Systematic Literature Review (SLR) to achieve a consolidated understanding of TD and determine its positive and negative outcomes. The study identified code decay and architectural deterioration as the major TD symptoms. The authors proposed a theoretical framework to help uncover TD elements by establishing boundaries and identifying the causes behind accruing TD. Budget and resource constraints were cited as potential causes of accumulating TD, and negative consequences in scheduling, risks, and quality were found as the associated outcomes.

Ampatzoglou et al. [21] conducted an SLR focusing on the financial perspective of TD. The authors found that the most common financial terms used in TD research were *principal* and *interest*. In contrast, the financial strategies that have been more frequently applied for managing TD are real options, portfolio management, cost/benefit analysis, and value-based analysis. The authors also emphasized that such strategies lacked consistency, i.e., the same strategy was differently applied in many studies. Sometimes, it lacked an explicit mapping between financial and Software Engineering concepts.

Li et al. [10] performed a Systematic Mapping Study (SMS) to understand TD and its management better. The

Table 2
Intelligent Techniques

Intelligent Technique	Description	Examples
Machine learning	Consists an evolving branch of computational algorithms designed to emulate human intelligence by learning from the surrounding environment. [13].	Regression and Support Vector Machine
Reasoning under uncertainty	It has been studied in the fields of probability theory and decision theory. This technique can be used for determining what is true in the world based on observations of the world [14].	Bayesian Network and Fuzzy logic
Search and optimization	Iterative computation process composed of encoding, initialization, selection, evaluation, and stop decision. They implement adaptive algorithms with potential learning opportunities [15].	Genetic algorithm, Linear programming
Natural language processing	Automatic text generation is associated with text processing, machine translation, speech synthesis and analysis, grammatical analysis, and style [16].	Speech recognition and text mining
Mathematical Model	An umbrella classification term for techniques that are not any of the other types and describes a system by variables and equations that establish relationships between the variables [11].	Conceptual model and Moving average
Multi-agent systems	Are systems comprised of agents who decide what to do based on their utilities. The agents can act autonomously, each with its information about the world and the other agents [14].	BDI and Cognitive Systems
Multiple decision criteria analysis	Is a sub-discipline of operations research that explicitly considers multiple criteria in decision-making problems [11].	Analytic Hierarchy Process (AHP)
Rules	It consists of rule-based expert systems, in which production rules represent the knowledge (i.e., if - then expressions) [14].	Decision tree
Recommendation systems	Computer-based systems that would work as a human expert. This technique is associated with processing knowledge and adopting complex decisions [17].	Recommenders
Semantic Network	A semantic network or net is a graph structure used for representing knowledge in patterns of interconnected nodes and arcs [18].	Complex networks and Graphs
Cognitive Simulation	Cognitive simulations are runnable computer programs representing models of human cognitive activities [19].	Data analytic and comprehension model

authors identified different TD types and TDM activities. They pointed out that the term "debt" has been used in different ways by different people, which leads to ambiguous interpretations of the term. The authors found that code-related TD and its management have gained the most attention. Finally, they concluded there is a need for more empirical studies with high-quality evidence on the whole TDM process and on the application of specific TDM approaches in industrial settings.

Alves et al. [22] conducted an SMS to investigate strategies proposed in the literature to identify and manage TD. The authors identified various TD indicators (i.e., God Class and Duplicate Code), which support identifying specific TD types. Software visualization techniques were the least used to identify TD. Regarding TD management strategies, the authors identified the portfolio approach and cost-benefit analysis as the most frequently cited strategies.

Lastly, Behutiye et al. [3] conducted a study to analyze and synthesize the state of the art of TD and its causes, consequences and management strategies in the context of Agile Software Development (ASD). They identified five research areas of interest related to the literature on TD in ASD. Among those areas, managing TD in ASD received significant attention, followed by investigations regarding software architecture in ASD and its relationship with TD. Moreover, eight categories regarding the causes and five categories regarding the consequences of incurring TD in ASD were identified.

3.2 TDM supported by Intelligent Techniques

Mostow [23], Partridge [24], and Ford [25] published the first studies on the topics and paved the road for this research area in the 80's. These authors compared Software Engineering and Artificial Intelligence (i.e., a type of Intelligent Technique), contrasting them in terms of the problems they attempt to solve and the methods, tools, and techniques used for both. These authors argued that a fusion of these two disciplines would be needed for many new demands and more advanced software solutions.

More recent studies have explored Intelligent Techniques and Technical Debt in more depth. For instance, Tsintzira et al. [26] conducted a literature review by analyzing the research corpus published in five high-quality SE journals with the goal of cataloging: (a) the software engineering practices in which machine learning (ML) is used; (b) the ML technologies that are used for solving them; and (c) the intersection of the ML and TDM: developing a problem-solution mapping. Similarly, Azeem et al. [27] presented an SLR on Machine Learning Techniques for Code Smell Detection. The authors focused on four different perspectives: (i) code smells considered, (ii) setup of machine learning approaches, (iii) design of the evaluation strategies, and (iv) a meta-analysis of the performance achieved by the models proposed so far. The analyses performed show code smells types and machine learning in the spotlight. In addition, the results also revealed several existing issues and open challenges that the research community can focus on in the near future.

3.3 Our Contribution

The amount of secondary studies on the topic of TDM shows its relevance for the research community and industry. Further, the TDM research community is following the trend of applying Intelligent Techniques to solve Software Engineering problems [5], [28], [29]. However, the existing secondary studies focus on how TDM is supported by ML techniques, ignoring other Intelligent Techniques such as fuzzy systems, genetic algorithms, and natural language processing. Additionally, these studies focused on specific TD subtypes (i.e., code smell) rather than analyzing TD more broadly in its various types and classifications. Furthermore, they do not detail which TD types or TDM activities are most common and their association with Intelligent Techniques.

Our study might be the pioneer in covering this gap and pointing out new and future research directions. The main differences between our study compared to the existing literature are: (i) we used a systematic approach for obtaining and analyzing studies related to Intelligent Techniques to support TDM activities; (ii) we sought to identify the intersection of a particular area of Software Engineering (i.e., TDM activities) with a broader concept of Intelligent Techniques; (iii) we researched the application of Intelligent Techniques (in addition to Machine Learning and other AI-Based techniques) demonstrating the need for a broader concept to define this category of techniques, (iv) we demonstrated the risk behind using Intelligent Techniques through a three-faceted analysis, involving the automation level, explanation level and point of application following the taxonomy provided by Feldt et al. [30], and (v) we assessed the maturity of research intersecting both areas by cataloging the empirical research types, contributions, and validation strategies described in primary studies.

4 Research Methodology

This section describes our research protocol. We conducted a Systematic Mapping Study (SMS) because we expected existing research to be fragmented and not follow common terminology or use theoretical concepts. Therefore, the SMS approach works well in such circumstances since it involves categorizing and aggregating knowledge dispersed across disconnected studies. Our approach is based on the guideline described by Petersen et al. [6].

4.1 Research Questions

Based on our research purpose - *to identify, classify, and analyze the use of Intelligent Techniques to support Technical Debt Management (TDM) activities* - the Research Questions (RQ) for this SMS are as follows:

RQ1. *What is the state of the art on the intersection of TDM and Intelligent Techniques?* Our first RQ focused on identifying what (i) Intelligent Techniques, (ii) TDM activities, and (iii) TD Types are in the spotlight. For this purpose, we derived three secondary RQs as follows:

— *RQ1.1.* What Intelligent Techniques have been employed to support TDM activities?

— *RQ1.2.* What TDM activities have been supported by Intelligent Techniques-based solutions?

— *RQ1.3.* What TD types have been supported by Intelligent Techniques-based solutions?

RQ2. *What is the risk level associated with Intelligent Techniques-based solutions for TDM?* Our second RQ focused on assessing the risks involved in the use of existing solutions that apply Intelligent Techniques to support TDM activities. For doing so, we used the taxonomy presented by Feldt et al. [30], which is composed of three facets: level of automation (i.e., from 1 to 10); explanation level (i.e., symbolist, analogizer, evolutionary, connectionist and Bayesian); and point of application (i.e., process, product, and runtime). According to Feldt et al. [30], in general, the higher the level of automation, the lower the explanation level, and as we move from the point of application being from the process to the runtime level, the riskier it is. The reasoning is that these situations mean having less control from humans, and it becomes costlier to reverse incorrect actions closer (or after) deploying the product. We argue that this information is vital for organizations in analyzing the feasibility of using these solutions and creating strategies for their adoption. Section 4.4 presents more details about the adopted taxonomy. We have derived three secondary RQs to address each of the facets separately:

— *RQ2.1.* What are the Automation Levels of Intelligent Techniques-based solutions for TDM?

— *RQ2.2.* What are the Points of Application of Intelligent Techniques-based solutions for TDM?

— *RQ2.3.* What are the Explanation Levels of Intelligent Techniques-based solutions for TDM?

RQ3. *What is the maturity of the existing Intelligent Techniques-based solutions for TDM?* Our third RQ focused on analyzing aspects of the empirical study conducted to develop and validate the proposed solutions. Such aspects include the empirical research types (i.e., experiment, observational study, experience report, case study, or systematic review) [31]; the research contribution types (i.e., procedure or technique, report, qualitative or descriptive model, tool, or empirical model) [32]; and research validation strategy (i.e., analysis, evaluation, example, experience, persuasion or blatant assertion) [32], which include the artifacts used (e.g., source code, bug report, and change report) [22] and the context of the application (i.e., open source, industrial, not specified). To this end, we derived three secondary RQs to address each of these aspects separately:

— *RQ3.1.* What empirical research types have been used by studies that reported Intelligent Techniques-based solutions for TDM?

— *RQ3.2.* What research contributions have been provided by studies that reported Intelligent Techniques-based solutions for TDM?

— *RQ3.3.* How the Intelligent Techniques-based solutions for TDM have been validated?

4.2 Search Method

Given the broad nature of the search required in this study, we used a hybrid search method. First, we performed a database search. Later, we used the primary studies identified in the database search as the seed set for a snowballing search (backward and forward) [33].

Database Search. When designing the search string, our main goal was not to miss relevant studies. Thus, we prioritized conducting a broader search with the expense of possibly having more studies filtered manually by applying the selection process. We defined the search string using the Population, Intervention, Comparison, and Outcome (PICO) criteria [34] as a reference. We focused on the *Population* and *Intervention* aspects, and ignored the *Comparison* and *Outcome* because they were out of our scope. We defined the *Population* facet with search terms about TD types and TDM activities. Similarly, we defined the *Intervention* facet using keywords that refer to Intelligent Techniques, based on the

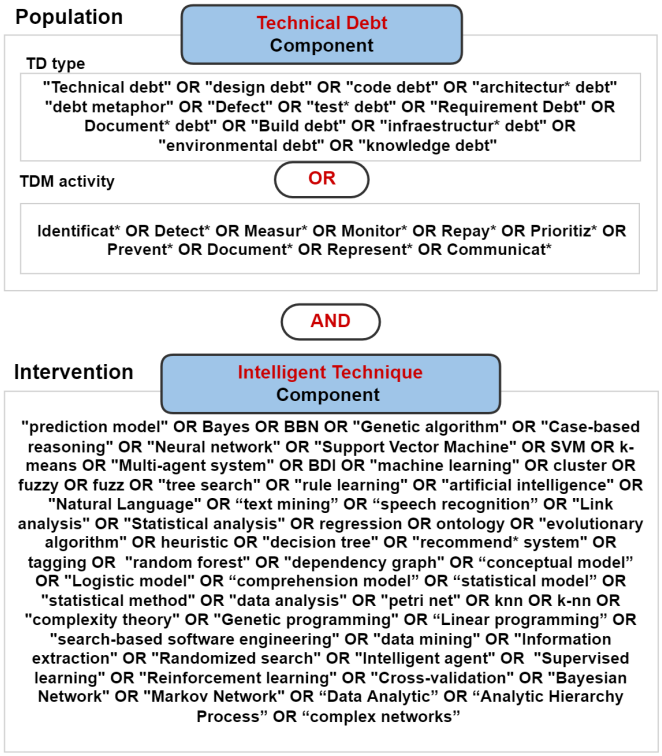

Figure 1. Search String.

search string presented in Perkusich et al. [5]. Figure 1 shows the final version of the search string employed in this SMS.

Moreover, we used Scopus [1], ACM [2], IEEE Xplore [3], and Engineering Village [4] as the data sources. The rationale behind selecting these databases was their extensive list of articles, journals, and conference proceedings related to Software Engineering. In addition, these databases were widely used in secondary studies related to our study [10], [3], [22], [5]. For each database, we executed tailored search queries given the required syntax.

We initially tested our string by applying it to the data sources selected for the study to check if it would return 14 known relevant papers, which we identified by manually exploring the proceedings of three conferences (i.e., MTD [5], TechDebt [6], and ICSME [7]) between the years 2010 and 2020 before conducting the SMS. As a result, all the papers were returned, indicating that the search string was comprehensive.

However, while conducting the study, we refined the terms related to the *Population* facet by adding synonyms (e.g., we added *Identification* as a synonym of *Detection*) and removing redundant terms (e.g., using the term "technical debt" as a prefix for the TDM activity terms).

Snowballing Search. We used Google Scholar for forward snowballing. As the seed set for the first iteration, we used the primary studies identified in the database search. In subsequent iterations, we analyzed only the newly primary studies identified in the preceding iteration. This process ended at an iteration where no new primary studies were identified.

1. https://www.scopus.com/
2. https://dl.acm.org/
3. https://ieeexplore.ieee.org/
4. https://www.engineeringvillage.com/
5. https://dblp.org/db/conf/icse-mtd/index
6. https://2020.techdebtconf.org/series/TechDebt
7. https://conferences.computer.org/icsm/

4.3 Selection Process

For the database search, we applied a three-steps approach to select the papers. First, we did a *preliminary screening* by applying a generic exclusion criteria:

1) A duplicate OR
2) Published in a non-peer reviewed channel (e.g., thesis) OR
3) Published in a book OR
4) Unavailable in English OR
5) Published before 2012 or after 2021.

We applied this step with the support of START tool [35]. Next, we applied an *RQs filtering* where we analyzed the remaining papers relevance in light of the study's RQs through the following inclusion criteria:

— Reports Intelligent Techniques-based solutions to support TDM activities AND
— A primary study.

For this purpose, we employed the adaptive reading approach [36], in which at least two reviewers evaluated each paper. Notice that the inclusion criteria only consider studies that used Intelligent Techniques to support one or more TDM activities. For example, since the terms "statistical model" and "statistical method" were in the search string, irrelevant papers were returned from the database search because they have applied some "statistical analysis" for validating hypotheses in studies not related to Intelligent Techniques. Consequently, we did not include such papers in our SMS because they did not satisfy the inclusion criteria.

Finally, we noticed that a few studies published their results in multiple papers. Thus, to minimize bias in our results, as the final step of the selection process, we only included the most comprehensive paper about the given study (i.e., *Removal of redundant papers*).

For the snowballing search, we applied the same three-steps approach previously described. One reviewer removed the papers given the generic exclusion criteria. Later, at least two reviewers applied the adaptive reading approach to select studies given the inclusion criteria. Finally, we kept only the most comprehensive paper about the given study for the cases in which a study was published in multiple papers.

We evaluated the reliability of the selection process by performing a pilot study on 20 randomly selected studies. For this purpose, two evaluators (the first and second authors) applied the followed the selection process independently on the 20 randomly selected studies. Once the selection process was validated, we applied it to all the papers retrieved by our search.

Results of the Selection Process. Figure 2 summarizes the retrieved papers in each step of the selection process. Next, we detail the procedures and results for each step.

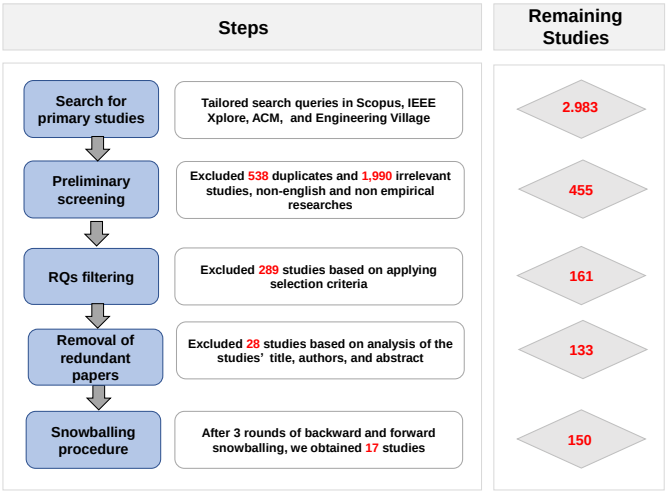

Figure 2. Summary of the SMS search and selection process.

1) *Search for primary studies*. We performed the search queries on 5/5/2022. As a result, we retrieved 2,983 papers. We extracted the papers' details in structured files (i.e., BibTex files) and inputted them into the START tool.

2) *Preliminary screening*. First, we used the START tool's features to remove papers given the generic exclusion criteria. As a result, we removed 2,528 (i.e., 538 duplicates and 1,990 irrelevant) papers, leaving us with 455 papers.

3) *RQs filtering*. We applied the adaptive reading approach to the 455 papers and removed 289 papers, leaving us with 161 papers.

4) *Removal of redundant papers*. We analyzed the 161 papers and identified the ones that were related to a single study by analyzing, for each paper, the list of authors, title, and abstract. As a result, we identified 28 primary studies reported in multiple papers. We only kept the most comprehensive paper for each study, leaving us with 133 primary studies.

5) *Snowballing procedure*. We used the 133 primary studies as the seed set for the first iteration of the snowballing search. We conducted the first iteration's search on 6/6/2022, retrieved 1,579 papers, and selected 12 primary studies. We conducted the second iteration's search on 6/16/2022, retrieved 264 papers, and selected 5 primary studies. Finally, we conducted the third iteration's search on 6/29/2022, retrieved 103 and selected zero primary studies, which indicated that this was our last iteration. As a result, at the end of the snowballing search, we included 17 primary studies, resulting in 150 primary studies in this SMS (Appendix I).

It is worth noting that about 15% of the primary studies were retrieved during the snowballing procedure, which shows the relevance of using snowballing to complement database search. The Supplementary Material presents the complete list of primary studies of our SMS [37].

4.4 Data Extraction and Synthesis

Once we identified all the primary studies, we extracted data regarding papers' demographics and research questions. Notice that we did not remove papers after the data extraction was initiated.

First, we created a data extraction sheet based on the information presented in Table 3. To ensure the reliability of the data extraction process, we piloted it by applying the process to 20 randomly selected primary studies. After the disagreements were resolved, we randomly assigned the remaining 130 primary studies to the researchers, where each study was assigned to a single researcher. Each researcher read the full text of the assigned primary studies to fill the data extraction sheet. Next, we describe the properties (Pr) shown in Table 3.

Paper's demographics. For *Pr1-Pr7*, we gathered data about relevant descriptive statistics and that could help us to detect studies reported on multiple papers.

RQ1. For *Pr8* and *Pr9*, we used the classification presented in Perkusich et al. [5]. For *Pr9*, we initially only used subcategories for Machine Learning, Reasoning under uncertainty, and Search and optimization techniques because they were the only ones further decomposed by Perkusich et al. [5] since they were the most prominent Intelligent Techniques identified in their study. While conducting our study, we did not see the need to decompose the remaining Intelligent Techniques into subcategories. For *Pr10*, *Pr11*, and *Pr12*, we reused other classifications widely employed by other secondary studies in TD research [10], [22] and [3].

RQ2. For *Pr13*, we classified each study using an integer scale between 1 and 10, where the higher the level of automation, the larger the value. These levels are based on the ten levels of automation of Sheridan-Verplanck [38], an existing taxonomy from Automation/HCI research focusing on human-computer decision-making. For *Pr14*, we used the framework proposed by Feldt et al. [39] that considers when and on what the Intelligent Technique was applied to the software system. The *process* level indicates that the Intelligent Technique-based solution was applied in the software development process and did not necessarily directly affect the source code. Examples here include solutions for TD prioritization. In contrast, the *product* level indicates that the solution directly affects the code, which is the case for automated program repair, e.g., automatically fixing a code smell. Finally, the *runtime* level affects the deployed product during runtime and has autonomous and self-adaptive systems as examples. For *Pr15*, we used the "five tribes of Artificial Intelligence" classification introduced by Domingos [40] and adapted by Feldt et al. [30]. Appendix A, B, and C details, respectively, the concepts of *Pr13*, *Pr14*, and *Pr15*.

RQ3. For *Pr16*, we used the classification of empirical study types presented by Tonella et al. [31]. For *Pr17* and *Pr18*, we used the classification schemes presented by Shaw [32]. Appendices D, E, and F detail such classification schemes. For *Pr19*, we used a catalog of the artifact proposed by Alves et al. [22] (i.e., *Source code*, *Documentation*, *Test artifacts*, *Bug report*, *System Architecture Specification*, *Backlogs Commit*, *Change Report*, *Requirement Specification*, and *Other*). Finally, for *Pr19*, we classified the context of the application

Table 3
Overview of the extracted data.

ID	Property	Format/value	RQ
Pr1-Pr4	Publication ID, title, abstract, keyword	Number, phrase, text, and set of words	-
Pr5-Pr7	Publication Year, source, metadata	Number, conference/journal, and data (e.g., URL, Volume, Pages, DOI, ISSN)	-
, Pr8	Intelligent Technique	Reasoning under uncertainty, Search and optimization, Machine learning, Mathematical model, Multi-agent system, Multiple criteria decision analysis, Rules, Recommendation system, Natural Language Processing, Semantic network, Cognitive simulation (See Table 2)	RQ1
Pr9	Intelligent Technique subcategory	Machine Learning (e.g., Regression Analysis, Neural networks, and Support Vector Machine), Reasoning under uncertainty (e.g., Bayesian network and fuzzy logic), and Search and optimization (Genetic Algorithm, Hill Climbing, Linear programming).	RQ1
Pr10	TDM Activity supported by Intelligent Technique	TD identification, TD measurement, TD monitoring, TD prioritization, TD communication, TD prevention, TD documentation, and TD repayment.	RQ1
Pr11	TD types supported by Intelligent Technique	Requirement TD, Architectural TD, Design TD, Code TD, Test TD, Build TD, Documentation TD, Infrastructure TD, Defect TD	RQ1
Pr12	TD subtypes supported by Intelligent Technique	Design TD (Code smells, Complex classes or methods, Not specified), Requirement TD (over-engineering, Not specified), Architectural TD (Architecture smells, Violations of good practices, Architectural compliance issues, System-level structure quality issues), among others.	RQ1
Pr13	Automation Level	1 to 10	RQ2
Pr14	Point of Application	Product, Process, Runtime	RQ2
Pr15	Level of Explanation	Symbolist, Connectionist, Evolutionary, Bayesians, Analogizers	RQ2
Pr16	Empirical Research Type	Experiment, Observational study, Experience Report, and Case study	RQ3
Pr17	Research Contribution	Procedure or technique, Report, Qualitative model, Analytic model, Tool or notation, Specific solution prototype, and Empirical Model	RQ3
Pr18	Validation Strategy	Analysis, Evaluation, Experience, Example, Persuasion, Blatant Assertion	RQ3
Pr19	Type of Artifact used for validating the study	Source code, Documentation, Test artifacts, Bug report, Architecture Specification, Backlogs, Change Report, Requirement Specification, Other	RQ3
Pr20	Validation Environment	Open source, Industrial, Not Specified	RQ3

of the proposed solutions as being for *open source* projects (i.e., users and stakeholders can adapt and modify them even for commercial purposes) , as part of an initiative with *industrial* goals (i.e., sale, modification, distribution, and adaptation of them must be authorized by the respective creator, being subject to legal punishment to break these rules), or not specified (i.e., the study did not provide enough data to classify the validation environment).

5 Results and Discussions

This section presents the gathered data and discusses the study's RQs. Section 5.1 summarizes demographic information of the primary studies. The following sections describe the results and discussions about the study's RQs. Due to space limitations, we provided a Supplementary Material containing the results not included herein (see Appendixes A to I [37]).

5.1 Demographic Information

Figure 3 illustrates the number of primary studies related to this study's topic from 2012 to 2021, showing an increasing trend. The number of studies on the topic has increased significantly in the past three years, indicating that it is becoming a hot research topic.

Further, 25% of the primary studies from conferences were published on MTD, TechDebt, and ICSE. Moreover, 30% of the primary studies from journals were published on TSE and JSS. Finally, eleven researchers were involved in more than four primary studies each, of which five of the most active authors are actively investigating the intersection of Intelligent Techniques and TDM since 43 of the 150 primary studies had at least one of these authors involved.

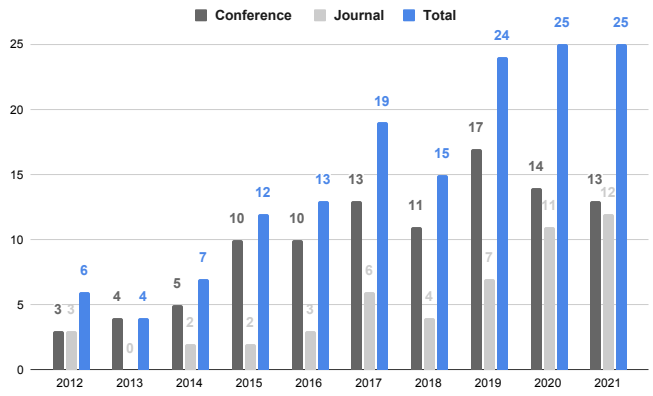

Figure 3. Frequency of primary studies published between 2012 and 2021.

5.2 Intersection Between TDM and Intelligent Techniques (RQ1)

Next, Sections 5.2.1, 5.2.2, and 5.2.3 summarize the data gathered for RQ1.1, RQ1.2, and RQ1.3, respectively. Finally, Section 5.2.4 analyses the results of RQ1.1, RQ1.2, and RQ1.3 in combination to address RQ1.

5.2.1 Intelligent Techniques Employed to Support TDM (RQ 1.1)

Figure 4 presents the frequency in which the Intelligent Techniques were used in the primary studies. The first paper on the topic of this study was published in 2012 and the most popularly applied Intelligent Technique for TDM is *Machine Learning*, identified in 47% of the primary studies. Out of these studies, approximately 60% of the Machine

learning-based solutions were classified as *Supervised Learning* techniques such as Neural Networks (28%), Regression Analysis (23%), and Support Vector Machine (about 8%). Further, approximately 40% of the studies used *Tree*-based techniques such as Decision Tree (about 21%) and Random Forest (14%).

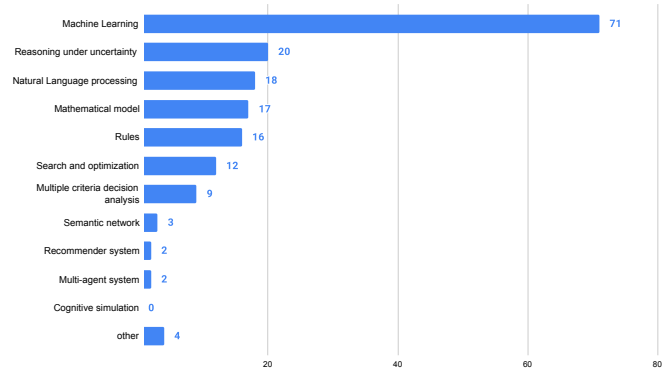

Figure 4. Distribution of primary studies by Intelligent Technique.

The second most frequently applied Intelligent Technique is *Reasoning Under Uncertainty*, identified in 13% of the studies. Out of these studies, approximately 75% of the solutions used *Statistical Model*, 35% used *Bayesian Network*, and 10% used *Fuzzy logic*.

The third most applied Intelligent Technique for TDM is *Natural Language Processing*, used by 12% of the primary studies, which was used to process artifacts produced during software development automatically. Next, we had *Mathematical Models* that was used for two main purposes. First, along with *Rules*, it was used to propose parameters and heuristics to detect data patterns in the software development artifacts [41], [42], [43]. Second, along with *Search and optimization*, *Mathematical Models* was applied to provide mechanisms for optimizing values for a given set of data or make predictions [44], [45], [46].

In summary, most of the Intelligent Techniques-based solutions for TDM focus on minimizing the effort of analyzing software development artifacts to detect trends and patterns. Such a motivation justifies the prominence of the use of *Machine Learning*, *Statistical Model*, and *Natural Language Processing* over techniques such as *Bayesian networks*, which, in software engineering, are mainly constructed with the assistance of domain experts [47], [48].

5.2.2 TDM Activities Supported by Intelligent Techniques (RQ1.2)

Table 4 shows the TDM activities that are supported by Intelligent Techniques-based solutions. Note that the total can exceed 100% since the same study can provide support to more than one TDM activity. The results show that most studies investigated *TD identification*, *TD measurement* and *TD monitoring*, which corresponds to a distribution of 65%, 26%, and 25% of the primary studies, respectively. Moreover, since the artifacts used for these TDM activities are formed mostly by (semi-) structured documents (e.g., source code, requirements, and architecture specification), the Intelligent Technique can be readily applied to gather important information for decision making. In contrast, *TD representation/documentation* and *TD communication* have received the least attention of primary studies, with only 3% and 2%, respectively.

Comparing these results with the findings of Li et al. [10], where the authors explored the most recurrent research topics in TDM, it is worth observing that they pointed out TD repayment (63%) as the hottest topic. However, by analyzing our results, only 11.3% of the Intelligent Techniques-based solutions focus on TD repayment, which does not follow this trend pointed out by the authors. Conversely, they also pointed out TD identification (54%) and TD measurement (52%) as the following hottest TDM activities, and they are the two most popular applications of Intelligent Techniques-based solutions.

Thus, we noticed that most studies that applied Intelligent Techniques-based solutions for TDM focused on the initial stages of the management process (e.g., TD identification and measurement). The infancy of the research topic can justify this trend. As the research area matures, it is expected to have more studies focusing on subsequent phases of the TDM process, including TD prevention.

Table 4
TDM activities and the numbers of studies.

TDM activity	No. of studies	%
TD identification	98	65,3
TD measurement	39	26
TD monitoring	37	24,7
TD repayment	17	11,3
TD prioritization	13	8,7
TD prevention	11	7,3
TD representation/documentation	5	3,3
TD communication	2	1,4

Finally, approximately 94 studies (about 65%) provided support for *only one* TDM activity, 43 studies (28%) provided support for *two* TDM activities. Less than 10% of studies provided support for *three or more* TDM activities. Therefore, it is still challenging to automate decision-making, considering the interconnection between all TDM activities in real-world software projects.

5.2.3 TD Types Supported by Intelligent Techniques (RQ1.3)

We decomposed the TD types given the root causes [10]. For the sake of consistency, we used the same classification mentioned in Section 5.3.2. When classifying the TD sub-types based on the causes of TD, if no reasons for a TD type were explicitly specified in a study, we classified the corresponding TD sub-type as "not specified". Figure 5 shows the resulting classification tree.

Design TD is the most popular TD type, being the target of about 53% of the primary studies. Following it, *Code TD*, *Architectural TD*, and *Defect TD* are explored in 33%, 21%, and 19% of the primary studies, respectively. On the other hand, *Requirement TD*, *Infrastructure TD*, and *Build TD* were the less mentioned TD types. Note that the total can exceed 100% since the same study can provide support to more than one TD type. Furthermore, it is important to mention that a few studies [49], [50], [51] did not clearly describe the TD type they were targeting.

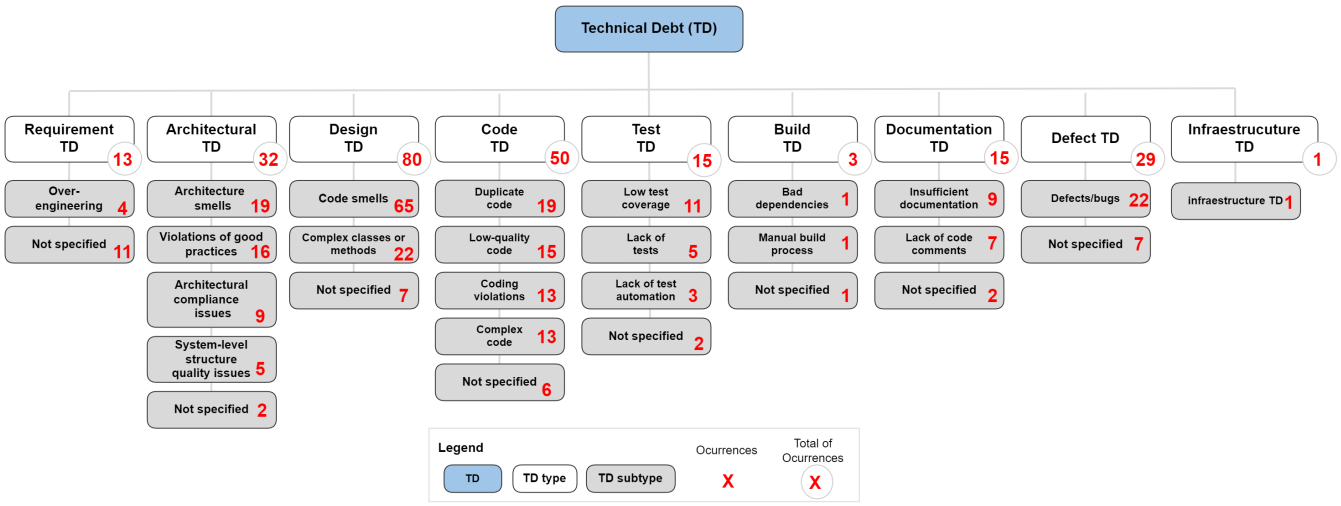

Figure 5. TD classification tree.

By comparing these results against other secondary studies, we can obtain an important comparative reference on the present research findings. First, Alves et al. [22] pointed out Design TD, Architectural TD, and Documentation TD showed up 20%, 24%, and 13%, respectively. Similarly, Li et al. [10] showed Code TD, Architectural TD, and Design TD with 80%, 65%, and 60%, being their results aligned with this SMS. Regarding the most studied TD types (top-3), these results are similar to previous studies reported in the literature. However, when analyzing the top-5 most studied TD types, we noticed differences compared to these aforementioned secondary studies. For instance, Test TD and Documentation TD received great attention, mainly in the last four years, representing about 20% of the primary studies considered in this SMS.

Another perspective concerns the TD type and its temporal progression. By analyzing the most consistently studied TD types are *Design TD* and *Code TD*, showing up 100% and 80% of the considered time-frame, respectively. Similarly, *Architectural TD*, *Defect TD*, *Requirement TD*, and *Test TD* appeared in 70% of the years. We also observed that the focus on Design TD and Defect TD has significantly increased over the years.

An essential finding of this study is that 64% of the studies provide support for *only one* TD type. Moreover, around 19% of the studies provide support for the *two* TD types, while about 17% of the studies provide support for *three or more* TD types simultaneously. Therefore, we noticed that the academic community lacks systematic awareness to deal with several TD types through well-defined and interconnected activities. As previously mentioned, most research was limited to applying Intelligent Techniques to address only one TDM activity. One possible reason may be the complexity of collecting information on certain TD types and automating decision-making for more than one TDM activity. As illustrated in Figure 5, the most supported TD types are Design TD, Code TD, and architectural TD. These types are already widely covered and mapped by metrics that can support decision-making more assertively.

5.2.4 *Joint Analysis - Intelligent Techniques for TDM (RQ1).*

Figure 6 illustrates the intersection between how Intelligent Techniques have been applied for TDM. As previously discussed, *TD identification* was the TDM activity with the most research effort. By analyzing Figure 6, we can see that this activity was targeted along with all TD types, being Design TD, Code TD, and Architectural TD the most recurrent ones. In addition, all types of Intelligent Techniques explored herein were applied to *TD identification*.

Analyzing from the point of view of Intelligent Techniques, we noticed that *Machine Learning* and *Mathematical model* were employed to support all the TDM activities. Similarly, *Design TD* and *Code TD* were the TD types in the spotlight. They received at least one type of Intelligent Technique to support all TDM activities. Finally, we identified clusters between TD identification and Design TD (76 occurrences) and TD identification and Machine Learning (78 occurrences), indicating the most recurrently used combinations between (i) TDM activity vs. TD type, and (ii) TDM activity vs. Intelligent Technique. These combinations were used by about 35% of the solutions based on Intelligent techniques to support TDM activities considered in this SMS.

Knowing that the main aims of applying Intelligent Techniques to TDM have been to identify, measure, and monitor TD explains why data analytics techniques such as Machine Learning, Statistical Models, and Natural Language Processing have been heavily used in this research field. This aspect is specific to this field, given that other techniques such as Bayesian networks and Search and Optimization are widely used in Software Engineering [52], [5]. In general, Bayesian networks have been primarily used for managing software risks and Search and Optimization techniques for prioritization and estimation purposes. Thus, as research efforts focus more on applying Intelligent Techniques to other TDM activities, which include TD repayment, TD prioritization, and TD prevention, such techniques are expected to receive more attention from researchers. Table 5 summarizes the main findings associated with RQ1.

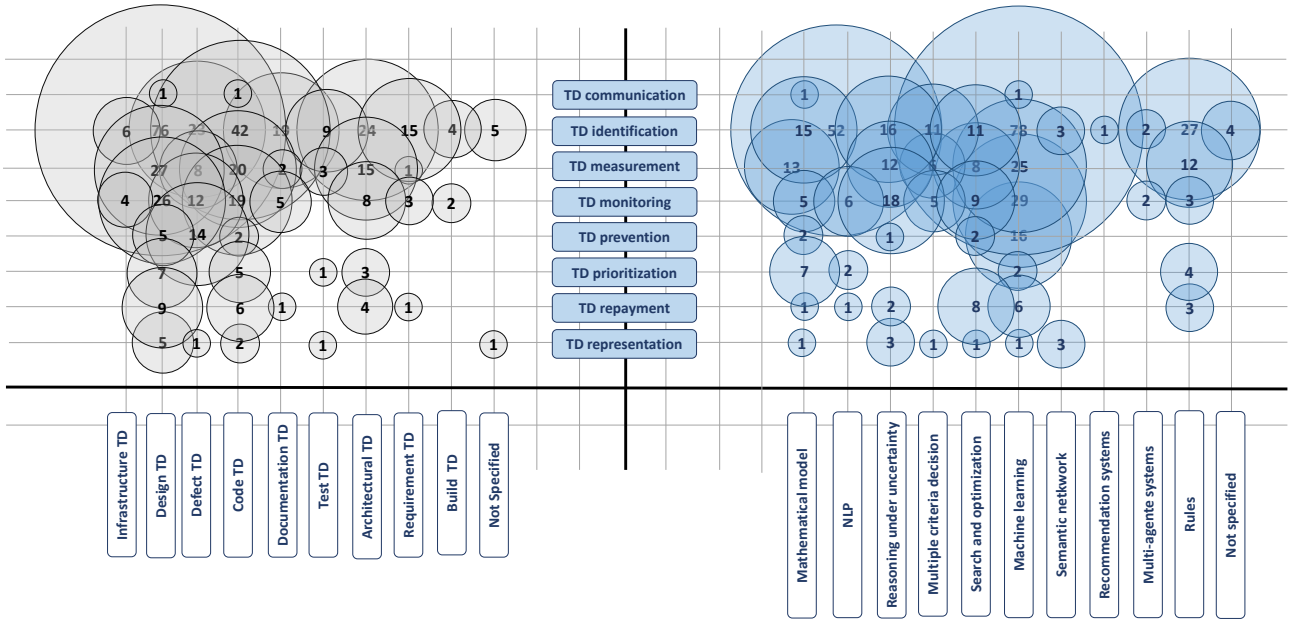

Figure 6. Relationship between TDM activities, TD types, and Intelligent Techniques.

Table 5
Takeaway Points - Intelligent Techniques for TDM (RQ1).

No.	Finding description
1	Machine Learning, Reason Under Uncertainty, and Natural Language Processing correspond to more than 70\% of the primary studies.
2	Around 85\% were concerned about presenting and validating a unique Intelligent Technique to support TDM activities.
3	The majority of the studies focused on TD identification (65\%), TD measurement (26\%), and TD monitoring (24\%).
4	No emphasis has been given to TD prevention and TD communication, which are crucial activities to avoid incurring TD.
5	The main TD types studied are Design TD (53\%), Code TD (33\%), and Architectural TD (21\%).
6	Most studies provided support for only one TD type (around 65\%). We also noticed a growing interest in using Intelligent Techniques that address two or more TD types, corresponding to 35\% of the studies.
7	Code TD and Design TD were explored considering all TDM activities, while other TD types were investigated for a maximum of four TDM activities.
8	Making a joint analysis of TDM activities, TD types, and Intelligent techniques, about 35% of the solutions provided support to TD identification for Design TD by using Machine Learning.

5.3 Risk Level Associated with the Intelligent Techniques-Based Solutions for TDM (RQ2)

Next, Sections 5.3.1, 5.3.2, and 5.3.3 summarizes the data gathered for RQ2.1, RQ2.2, and RQ2.3, respectively. Finally, Section 5.3.4 analyses the results of RQ2.1, RQ2.2, and RQ2.3 in combination to address RQ2.

5.3.1 Automation Level of the Intelligent Techniques-Based Solutions (RQ2.1)

Figure 7 illustrates the automation level (AL) of the Intelligent Techniques-based solutions for TDM. Notice that a few studies have achieved different levels of automation, depending on the targeted TDM activities and the employed Intelligent Techniques. The TDM activities targeted by the primary studies explain the distribution of automation levels shown in Figure 7.

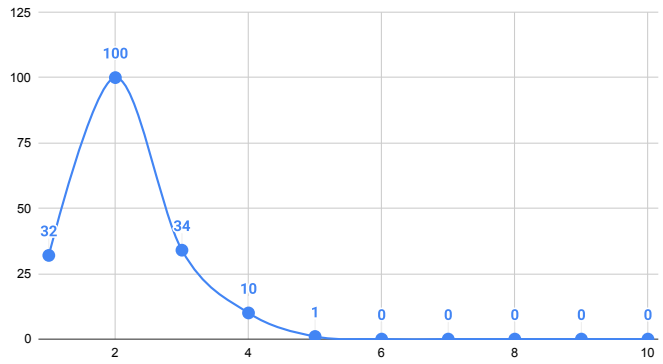

Figure 7. Automation Level of the Intelligent Techniques-based solutions for TDM based on the classification scheme proposed by Sheridan [38].

About 75% of the primary studies provided low levels (i.e., 1 and 2). The data indicate that most solutions focus on supporting decision-making by offering humans a set of alternatives. This result aligns with the fact that most studies targeted *TD identification*, *TD measurement*, and *TD monitoring*, which focus on gathering insights from the artifacts and not necessarily performing actions on them.

The studies that achieved higher levels of automation primarily focused on TD prioritization (40%) and TD repayment (30%). Logically, focusing on these TDM activities brings the opportunity to achieve high automation levels since they require actions to be taken.

Moreover, we noted that the solutions presented by the primary studies included in this SMS tend to have high AL when combining two or more Intelligent Techniques. The combined use of various Intelligent Techniques can

take advantage of the best features of each technique. For instance, a solution based on two Intelligent Techniques might use the first one (e.g., Natural Language Processing) to determine the number of options/actions and the second one (e.g., Machine Learning) to select the options/actions based on a set of criteria previously defined [53], [54].

5.3.2 Point of Application of the Intelligent Techniques-Based Solutions (RQ2.2)

Regarding the Point of Application (PA), most of the Intelligent Techniques-based solutions for TDM focus on the *process* (e.g., [55], [56], and [57]). By contrast, the smallest part of the Intelligent Techniques in the primary studies provides support to *Product* (8%) and *Runtime* (3%) PA facets. Note that the total can exceed 100% since the same solution can provide more than one PA simultaneously. This result can be explained by the nature of TDM, which is a management activity, and the TDM activities mostly targeted by the primary studies: *TD identification*, *TD measurement*, and *TD monitoring*, which do not necessarily include changing the source code or deployed software.

5.3.3 Explanation Level of the Intelligent Techniques-Based Solutions (RQ2.3)

The Explanation Level (EL) of the Intelligent Techniques-based solutions for TDM has the following frequency: Symbolist (about 37%), Analogizers (about 35%), Connectionist (about 17%), Bayesians (about 10%), and Evolutionary (about 6%). Note that the total can exceed 100% since a few primary studies reported using multiple Intelligent Techniques simultaneously.

Notice that only 17% of the proposed solutions used Connectionist techniques such as neural nets, which are hard to be analyzed and tested. The remaining solutions used techniques that can, at some level, have their logic analyzed before deployment, reducing their risks.

5.3.4 Joint Analysis - Risk Level Associated with Intelligent Techniques-based Solutions for TDM (RQ2)

Discussing the risks of deploying an Intelligent Techniques-based solution by analyzing each of the three facets individually can lead to erroneous conclusions. Thus, this section discusses such risks by combining the previously discussed insights for each facet.

Table 6 shows a three-faceted analysis between the PA, AL, and EL. By combining the data of all facets, we can conclude that most of the Intelligent Techniques-based solutions for TDM focused on the *process*, low AL, and high EL (i.e., solutions that are easy to be analyzed and tested). In conclusion, most of the identified solutions have low risk.

However, it is worth mentioning that risk assessment also depends on the context of the adopting organization. For example, in general, using a Connectionist technique for high AL is risky. Still, such risks might be reduced depending on the adopting organization's know-how and experience and the risk-reduction strategies adopted to trust the solutions. As another example, for certain well-understood domains, having higher levels of automation might be less risky than trusting humans to do the job. Table 7 summarizes the main findings associated with RQ2.

5.4 Maturity of the Intelligent Techniques-Based Solutions to Support TDM activities (RQ3)

Next, Sections 5.4.1, 5.4.2, and 5.4.3 summarizes the data gathered for RQ3.1, RQ3.2, and RQ3.3, respectively. Finally, Section 5.4.4 analyses the results of RQ3.1, RQ3.2, and RQ3.3 in combination to address RQ3.

5.4.1 Empirical Research Type of the Intelligent Techniques-Based Solutions(RQ3.1)

Figure 8 shows the distribution of the empirical research types employed by the primary studies, which influences the strength of the evidence. The most employed empirical research type was *Experiment*, used by around 40% of the primary studies. Next, *Case study* and *Experience report* were employed by, respectively, 28% and 23% of the primary studies. Finally, only 9% of the primary studies conducted *Observational studies*.

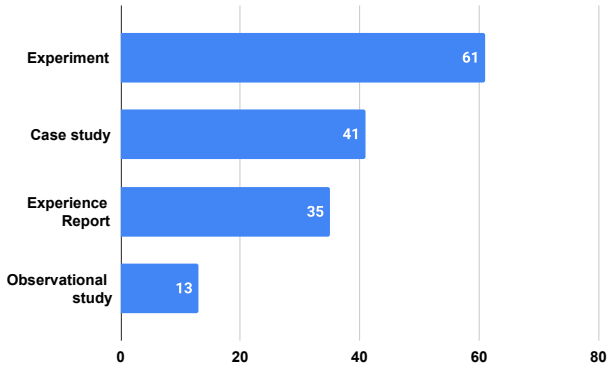

Figure 8. Publication distribution by Empirical Research types.

Having most of the studies focusing on *Experiments* and *Case studies* is a positive indicator of the strength of the evidence. However, given the scarcity of available Intelligent Techniques-based solutions for TDM, most of the *Experiments* focused on comparing the proposed Intelligent Techniques-based solution to conventional solutions. Further, most of the *Case studies* focused on evaluating the practical utility of a proposed solution without worrying about comparing it with competing solutions.

This information indicates that this area of research is still in its infancy. Ideally, for a scientific field to develop, researchers need to make their data and solutions available, enabling others to perform comparative studies [58].

5.4.2 Research Contribution of the Intelligent Techniques-Based Solutions (RQ3.2)

Figure 9 illustrates the distribution of the research contribution types of the primary studies. The most popular research contribution type is *Procedure or technique*, developed by around 35% of the primary studies, followed by *Analytic model* (25%), *Tool or notation* (13%), and *Specific solution* (10%).

We highlight the scarce availability of tools developed, which, as previously discussed, might hinder the scientific development of the field. A similar phenomenon was observed by Perkusich et a. [5] while exploring the use of Intelligent Techniques-based solutions for agile software development. However, the expectation is that as the research

Table 6
Automation Level vs. Point of Application (PA) vs. Explanation Level (EL).

	Automation Level															
	1			2			3			4			5			
PA / EL	Pc	Pd	Rt	Pc	Pd	Rt	Pc	Pd	Rt	Pc	Pd	Rt	Pc	Pd	Rt	Total
Analogizers	11			38	2	1	16	1		5		1				**75**
Bayesians	2			8	2		4			1						**17**
Connectionist	4			19	1		9	1		1						**35**
Evolutionaries	1			4			2			2		1	1			**11**
Symbolist	15	1	1	38	3		9	3		1	2	1				**74**
Total	**33**	**1**	**1**	**107**	**8**	**1**	**40**	**5**	**0**	**10**	**2**	**3**	**1**	**0**	**0**	**220**
Legend: Pc - Process, Pd - Product and Rt - Runtime																

Table 7
Takeaway Points - Risks of Using Intelligent Techniques to Support TDM Activities (RQ2).

No.	Finding description
1	About 88\% of the studies were categorized at AL 1 and 2. By contrast, about 29\% of studies had AL 3 and 4. Finally, only one study had AL 5.
2	Regarding PA, most studies were interested in Process (about 93\%). The smallest part of the studies was interested in Product (8\%) and Runtime (about 3\%).
3	Symbolist and Analogizers were the EL in the spotlight, showing up 37\% and 35\%, respectively. Less than 20\% of the primary studies used Bayesian and Evolutionary strategies.
4	Making a joint analysis of automation level, point of application, and level of explanation, about 35% of solutions presented 2 as AL, process as PA, and Symbolist or Analogizer as EL, respectively.
5	Using the AL, PA, and EL provides a basis for software engineers to consider the risks of applying Intelligent Techniques to support TDM activities.

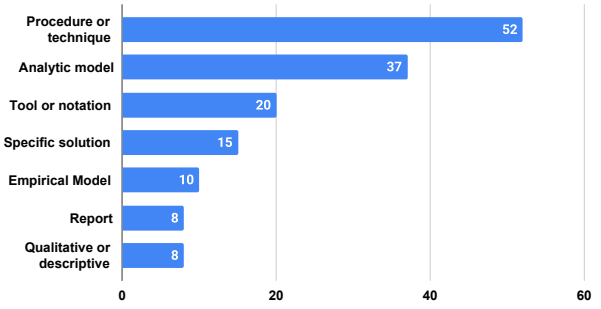

Figure 9. Publication distribution by Research Contribution types.

area matures, many of the *procedures or techniques* and the *Analytic Models* evolve into tools and are made available for the research community to validate.

5.4.3 Research validation of the Intelligent Techniques-Based Solutions (RQ3.3)

This section discusses the research validation strategies employed by the primary studies by focusing on the validation strategies [32], validation environment, and the artifacts used to validate the solutions. Figure 10 illustrates the distribution regarding the research strategies employed to validate the Intelligent Techniques-based solutions for TDM. The most common validation strategies were *Analysis*, *Evaluation*, and *Examples* used by 37%, 24%, and 21% of the primary studies, respectively.

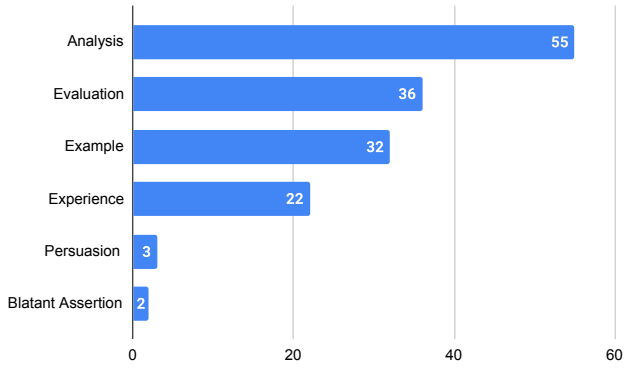

Figure 10. Publication distribution by Validation Strategies.

Regarding the validation environment, we observed the following distribution: *industrial* 20%, *open-source* 72%, and *not specified* 16%. The total exceeds 100% since the same study can use more than one validation environment. The data show the most studies focused on analyzing open-source projects, which has the advantage of eases data sharing.

Regarding the artifacts, we observed that the most used artifacts were *source-code* and its derived metrics (81%), *software architecture* (13%), and *software documentation* (13%). Combining these artifacts represents 95% of the artifacts used by the primary studies. Note that the total can exceed 100% since the same study can use more than one artifact.

Most studies (73%) used only one type of artifact; even though, in some cases, additional information is derived for TDM activities. Similarly, we observed that 22% of the analyzed studies made combined use of two artifacts (e.g., source code and architecture specification). In 5% of the cases, the authors used three or more artifacts.

5.4.4 Joint Analysis - Maturity of the Intelligent Techniques-based Solutions for TDM (RQ3)

Figure 11 illustrates the intersection between *Empirical Research Type*, *Research Contribution*, and the *Validation Strategies*. By analyzing it, we noted that the Empirical Research Type classified as *Experiment* was the most recurrent among the Research Contribution (6 out of 7) and Validation Strategies (4 out of 6) in quantitative terms. However, we can note

that the Empirical Research Type classified as *Experience Report* was used in all types of contribution (7 out of 7) and validated by all strategies (6 out of 6) considered in this study.

Further, we identified clusters between *Experiment vs. Analysis* (30 occurrences) and *Experiment vs. procedure or techniques* (32 occurrences), indicating the most recurrently used combinations between (i) Empirical Research Type vs. Research Contribution, and (ii) Empirical Research Type vs. Validation Strategies. These combinations were used by about 40% of the studies considered in this SMS. Table 8 summarizes the main findings associated with RQ3.

Table 8
Takeaway Points - Maturity of solutions (RQ3).

No.	Finding description
1	Experiment and Case study were the most recurrent Empirical Research types showing, respectively, up to 41\% and 27\% of the primary studies.
2	Procedure or technique, analytic model, and tool or notation were the most recurrent Research Contributions, showing up 35\%, 25\%, and 13\%, respectively, of the primary studies.
3	The most common Validation Strategies used in the primary studies were *analysis*, *examples*, and *evaluation* showing up 37%, 24%, and 21%, respectively.
4	Around 20\% of the primary studies were validated in an industrial environment. In contrast, most studies (about 72\%) used open-source projects to perform validation of their solutions.
5	Source code was the most common artifact used (81\%), followed by Software Architecture (13\%). About 73\% of the primary studies have used only one type of artifact to validate their solutions.
6	Making a joint analysis of solutions based on intelligent techniques, they used more recurrently Experiment (as Empirical Research Type), Procedure or technique (as a Research Contribution), and Analysis (as a Validation Strategy).

6 IMPLICATIONS

This section discusses the implications for the research community (Section 6.1) and industrial practitioners (Section 6.2).

6.1 Implications for researchers

For researchers, this study's findings point to the following implications:

— research on the application of Intelligent Techniques to support TDM activities is highly concentrated on a few TD types (e.g., design TD, architecture TD, code TD, and defect) while other types are currently under-investigated. This result shows a clear gap that could be explored in the coming years. A possible explanation for this is that a plethora of tools perform source code analysis and can be used to support the detection of TD from the source code. Moreover, it can be seen that the TD types have expanded with time, indicating that new fields are being included, such as infrastructure, build, and versioning.

— similar analysis can be done in research on TDM activities. We noticed that most primary studies concentrated on using Intelligent Techniques to support a few TDM activities (e.g., TD identification, TD measurement, and TD Monitoring), while other TDM activities are under-investigated, demonstrating the need for future investigations. For instance, although TD representation/documentation has been shown to benefit the process of software understanding, there is still little investigation relating to this activity. The researchers can investigate using Intelligent Techniques (e.g., Natural Language Processing and Repository Miner) to support these activities.

— there are different TD types and some indicators for each of them, but we have not identified any evidence on how to use this set of information to guide initiatives of TD management in real settings. Despite progress in different areas of TD, there is still a need to look at the big picture and investigate holistic strategies for managing TD effectively in the software industry. Investigating Intelligent Techniques in this context can be a promising alternative to automate and support decision-making on TDM activities.

— few empirical studies have been performed in an industrial environment. This result indicates that, for some areas, we still do not fully understand all the costs or benefits of the proposed TD indicators and management strategies. Many of these proposals require deep investigation. Some of them were just cited in some papers.

— there are four categories of Intelligent Techniques responsible for more than 70% of the primary studies: Machine Learning, Reasoning under uncertainty, Natural Language Processing, and Mathematical Models. On the other hand, no robust empirical evidence demonstrates the ineffectiveness of other intelligent techniques to support TDM activities. We highlight that, to advance this research area, there is a need to increase the research effort in other Intelligent Techniques types.

— the quality assessment also revealed that the average evidence level of the claims related to the application of Intelligent Techniques on TDM activities is only 0.60. This result means that we lack empirical studies with a high level of evidence, which could give TD stakeholders (e.g., architects, developers, and managers) more confidence in applying Intelligent Techniques-based solutions to various TDM activities.

— the notions of "TD types", "TDM activities" and "Intelligent Techniques" mean different things to different research communities, including Software Engineering, Artificial Intelligence, and others. This aspect was notorious mainly in studies published before 2015, representing a big challenge for our study. In the case of TD and TDM activities, there are some proposed classifications that we have noticed as being comprehensive and descriptive [10][3]. However, not all primary studies included adequately use such classifications. In the case of intelligent techniques, only recently the use of this term has been proposed in the sense of encompassing a set of techniques with similar characteristics (see Section 2.2). From this, we can conclude that researchers still struggle to establish a single taxonomy to classify the previously described items. With the growing

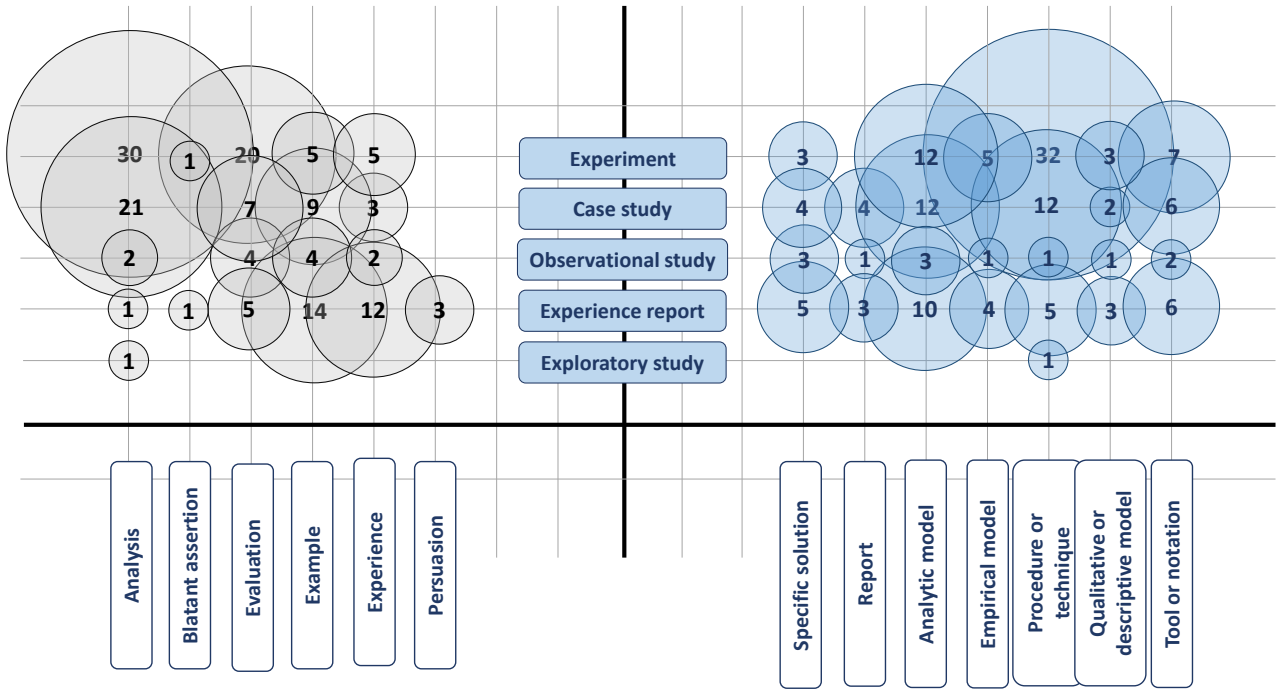

Figure 11. Empirical research type vs. Research contribution vs. Validation strategies.

number of secondary and tertiary studies, this gap is expected to be resolved soon.

Our results clearly show an active and fruitful area of investigation that is continuing to grow and still needs maturation in terms of consolidating concepts and empirically validating new solutions. For this body of work to present a valuable contribution to practice, the research community must find ways to guide practitioners to those Intelligent Techniques most likely helpful in a particular context and adapt those to a given situation.

6.2 Implications for practitioners

This study has primarily focused on facilitating and guiding future research in the application of Intelligent Techniques to support TDM activities. However, our results have important implications for industrial practitioners as well, particularly those looking to the literature for guidance on how to properly manage TD on real-world projects:

— this study lists innovative and promising studies that investigated applying Intelligent Techniques for TDM. It can be seen as a starting point for practitioners to analyze the available solutions and kick-start initiatives for using and evolving them.

— most of the existing solutions focused on the initial stages of the management process (e.g., TD identification and measurement). Thus, if practitioners wish to carry out a complete Intelligent Techniques-based TDM process, there is currently a lack of adequate solutions to support such initiatives.

— even though the risk level of adopting an Intelligent Technique-based solution depends on the context of the adoptee organization, the transparency in which we assessed the risks by considering AL, PA, and EL is an indicator to support practitioners in assessing the feasibility of adopting them.

— in general, if the Intelligent Techniques-based solution is new to the organization, we suggest that practitioners should start at low AL to allow more human intervention and at a "lower" PA, where potential issues with introducing the technique will not be directly introduced to the source-code. Similarly, it can be preferred to use Symbolist and Analogizers "EL classifications" since it is easier to understand the rationale of the results from these classifications compared to *Bayesian, Evolutionaries, and Connectionist*. Then, by building more experience, these solutions can expand to explore higher automation levels, new points of application, and a more complex level of explanations of the Intelligent Technique to support some TDM activity.

Our study indicates a need for a common research agenda for the field to progress and increase the impact on industrial practitioners. It is beyond the scope of this article to suggest such an agenda. However, since the field requires a plethora of skills, there is a need for researchers from different communities (e.g., TD, Machine Learning, Search and Optimization, Reasoning Under Uncertainty, Knowledge Management, Data Analytics, Software Metrics, and Decision-Support Systems) to define the basis from which the field can evolve. Furthermore, a more significant portion of solutions based on Intelligent Techniques for supporting TDM activities must be evaluated and validated in an industrial environment. This aspect could contribute to a greater understanding of the industrial practitioners' real needs and facilitate technology transfer from the academic community.

7 Limitations and Threats to Validity

The results of this study may have been impacted by the coverage of the search method (Section 7.1), bias on study selection (Section 7.2), inaccuracy of data extraction (Section 7.3), and bias on the data synthesis (Section 7.4).

7.1 Incompleteness of the search method

We aimed to retrieve as many relevant primary studies with the database search. We faced a challenge in determining the scope of our study as the notion of "TD types", "TDM activities" and "Intelligent Techniques" mean different things to different research communities, including Software Engineering, Artificial Intelligence, and others. Therefore, to cover them all and avoid bias, we searched the literature based on relevant terms, including secondary studies, and combined them in our search string (Figure 1).

We used TD types in the search string instead of specifying its subtype. This initiative would have led to a considerable increase in studies, making the process of extraction and selection of studies more complex and costly in terms of time and effort. For instance, if we added the term "code smell" to our string, thousands of studies associated with this TD subtype would have been retrieved, which could bias the selection process.

Another threat is related to missing relevant primary studies. By applying the search string, we obtained 2.445 different studies. Due to this large number of retrieved studies, we spent significant time and effort in the filtering and selection processes. After carrying out these processes, we obtained 133 primary studies. We also performed a snowballing search and identified additional 17 primary studies.

However, we highlight that it is unfeasible to completely eliminate the threat of missing relevant studies because of the broadness and immaturity of the area, which includes the use of inconsistent terminology, mainly when dealing with Intelligent Techniques.

We consider our sample relevant and representative of our study's aims as it provided an overview of application Intelligent techniques to support TDM activities. We seek to maximize the amount of work by choosing four relevant databases for research. The rationale behind selecting these databases was an extensive list of articles, journals, and conference proceedings related to Software Engineering and the Intelligent Techniques they provide. Additionally, these databases were widely used in secondary studies related to our study. These databases proved effective because they returned all the studies identified in the pilot study.

7.2 Bias in the study selection process

We mitigated the bias in the study selection by having each paper analyzed by at least two researchers following the adaptive reading approach. It is essential to mention that whenever there was a disagreement between two reviewers regarding the criteria for selecting/extracting the studies, a third reviewer was triggered to resolve such disagreement.

Further, we defined unambiguous inclusion and exclusion criteria for study selection. However, different researchers are prone to have different understandings of these criteria, so the selection results tend to be varied. We mitigated this threat by conducting a pilot study to ensure that the researchers reached a consensus on understanding the inclusion and exclusion criteria.

7.3 Inaccuracy of data extraction

We have adopted three measures to mitigate inaccuracies and bias in the data extraction and synthesis. First, the data items to be extracted were discussed among the researchers, and agreement on the meaning of each data item was achieved. Second, a pilot data extraction was performed among three researchers, and disagreements on the results of the pilot data extraction were discussed to reach a consensus. Third, at least two researchers checked the data extraction results to mitigate any discordance.

7.4 Bias on data synthesis

Further, we observed that not all papers sufficiently described the details of information (e.g., TD type, TDM activity, Intelligent Techniques) that we aimed to extract. Therefore, we had to infer certain pieces of information from data items during data synthesis. In other cases, the studies used different terminology. For example, some studies interchangeably used "quantifying" meaning "measuring" and "monitoring". We reduced the bias in the synthesis by using readily available classification schemes whenever possible (e.g., the TDM activities scheme proposed by Li *et al.* [10]) and by holding meetings whenever difficulties were encountered to elucidate these issues.

Finally, we mitigated bias on data synthesis by holding discussion and peer review of the extracted data by the researchers, having a structured template for data synthesis, and several steps where the scheme and process were refined and evaluated.

8 Final Remarks

This study provides a comprehensive view of research on Intelligent Techniques to support Technical debt Management (TDM) activities. It identifies, analyzes and classifies the published studies (between 2012 and 2021) in four electronic databases: IEEE Xplore, ACM, Scopus, and Engineering village. As a result, we identified 2,445 different studies, out of which we selected 133 primary studies. Later, we performed a snowballing search and identified additional 17 primary studies, totaling 150 primary studies.By analyzing the extracted data from these studies, we got a comprehensive understanding of the application of Intelligent Techniques to support TDM activities and an overview of the current state of the research in this area.

Our study findings can be summarized as follows:

— **RQ1**. Regarding the intersection of Intelligent Techniques and TDM, about 35% of the solutions focused on TD identification for Design TD using Machine Learning techniques. This Intelligent Technique was widely employed in all TDM activities. Similarly, Design TD and Code TD were explored considering all TDM activities, while other TD types were investigated for a maximum of four TDM activities. Most of the

presented solutions (about 90%) primarily support one or two TDM activities, denoting the lack of a more global vision of the TD management process. Besides, around 85% presented solutions using only one type of Intelligent Technique. Future research can explore other alternatives or combine intelligent techniques to compare results. Additionally, most solutions described in primary studies provided support for only one TD type (64%). We also noticed a growing interest in using Intelligent Techniques to address two or more TD types;

— **RQ2**. About 40% of solutions presented 2 as AL, being applied in the Process PA, and presented Analogizer as EL, respectively. We noticed that most Intelligent Techniques AL 1 and 2 supported TD identification (about 60%). This TDM activity requires a low level of automation since the task of inferring the existence (or not) of TD instances does not imply a high set of options. Regarding PA, this indicates that Intelligent Techniques were mainly applied in the software development process and did not necessarily directly affect the source code already deployed. Regarding EL, we note that the studies mostly used simpler techniques to analyze and test (e.g., Analogizers and Symbolists) compared to others (e.g., Connectonists). Finally, using the EL in combination with the AL and PA provides a basis for software engineers to consider the risks of applying Intelligent Techniques to support TDM activities; and

— **RQ3**. Regarding the maturity of the existing solutions, about 25% of the solutions were of the type *Procedure or technique*, used *Experiment* as the empirical research type, and validated their results through *Analysis*. Around 20% of the primary studies were validated in an industrial environment. In contrast, most studies (about 72%) used open-source projects to perform validation of their solutions. Source code was the most common artifact used (81%), followed by Software Architecture (13%). About 73% of the primary studies have used only one type of artifact to validate their solutions.

The results of the study can benefit researchers and practitioners. For researchers, our study provides a structured understanding of the state of the art of Intelligent Techniques within the context of TDM activities. They can base their studies on the topic and recognize research gaps that require further investigation. For practitioners, our research findings can better understand the use of Intelligent Techniques in the context of TDM activities. Furthermore, practitioners can use the mapping presented herein of how Intelligent Techniques have been used to support TDM activity as a starting point for improvement initiatives to automate their TDM decision-making processes. Moreover, there is a need for more empirical studies with high-quality evidence on the entire TDM process and applying specific Intelligent Techniques in industrial settings. Finally, more sophisticated and dedicated solutions based on Intelligent Techniques are needed for managing various TD types in the whole TDM process.

ACKNOWLEDGMENTS

This research is supported in part by the IFPB employee qualification incentive program (PIQIFPB) - Public Notice Nr 21/2021/PRPIPG.

REFERENCES

[1] P. Avgeriou, P. Kruchten, I. Ozkaya, and C. Seaman, "Managing technical debt in software engineering (dagstuhl seminar 16162)," in *Dagstuhl Reports*, vol. 6, no. 4. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

[2] A. Martini, T. Besker, and J. Bosch, "Technical debt tracking: Current state of practice: A survey and multiple case study in 15 large organizations," *Science of Computer Programming*, vol. 163, pp. 42–61, 2018.

[3] W. N. Behutiye, P. Rodríguez, M. Oivo, and A. Tosun, "Analyzing the concept of technical debt in the context of agile software development: A systematic literature review," *Information and Software Technology*, vol. 82, pp. 139–158, 2017.

[4] C. Seaman, Y. Guo, N. Zazworka, F. Shull, C. Izurieta, Y. Cai, and A. Vetrò, "Using technical debt data in decision making: Potential decision approaches," in *2012 Third International Workshop on Managing Technical Debt (MTD)*. IEEE, 2012, pp. 45–48.

[5] M. Perkusich, L. C. e Silva, A. Costa, F. Ramos, R. Saraiva, A. Freire, E. Dilorenzo, E. Dantas, D. Santos, K. Gorgônio *et al.*, "Intelligent software engineering in the context of agile software development: A systematic literature review," *Information and Software Technology*, vol. 119, p. 106241, 2020.

[6] K. Petersen, S. Vakkalanka, and L. Kuzniarz, "Guidelines for conducting systematic mapping studies in software engineering: An update," *Information and software technology*, vol. 64, pp. 1–18, 2015.

[7] D. Albuquerque, E. Guimaraes, G. Tonin, M. Perkusich, H. Almeida, and A. Perkusich, "Comprehending the use of intelligent techniques to support technical debt management," in *International Conference on Technical Debt (TechDebt'22)*, 2022.

[8] W. Cunningham, "The wycash portfolio management system," *SIGPLAN OOPS Mess.*, vol. 4, no. 2, p. 29–30, Dec. 1992. [Online]. Available: https://doi.org/10.1145/157710.157715

[9] V. M. Silva, H. J. Junior, and G. H. Travassos, "A taste of the software industry perception of technical debt and its management in brazil," *Journal of Software Engineering Research and Development*, vol. 7, pp. 1–1, 2019.

[10] Z. Li, P. Avgeriou, and P. Liang, "A systematic mapping study on technical debt and its management," *Journal of Systems and Software*, vol. 101, pp. 193–220, 2015.

[11] T. Das and D. Bebarta, "Intelligent techniques: An overview," *Intelligent Systems: Advances in Biometric Systems, Soft Computing, Image Processing, and Data Analytics*, p. 1, 2019.

[12] P. Kumar and S. Singh, "An emerging approach to intelligent techniques—soft computing and its application," in *Internet of Things and Big Data Applications*. Springer, 2020, pp. 171–182.

[13] E. Alpaydin, *Introduction to Machine Learning*, 3rd ed., ser. Adaptive Computation and Machine Learning. Cambridge, MA: MIT Press, 2014.

[14] D. L. Poole and A. K. Mackworth, *Artificial Intelligence: foundations of computational agents*. Cambridge University Press, 2010.

[15] A. Kumar, R. Nagar, and A. S. Baghel, "A genetic algorithm approach to release planning in agile environment," in *2014 International Conference on Information Systems and Computer Networks (ISCON)*. IEEE, 2014, pp. 118–122.

[16] E. da Silva Maldonado, E. Shihab, and N. Tsantalis, "Using natural language processing to automatically detect self-admitted technical debt," *IEEE Transactions on Software Engineering*, vol. 43, no. 11, pp. 1044–1062, 2017.

[17] Y. Duan, J. S. Edwards, and Y. K. Dwivedi, "Artificial intelligence for decision making in the era of big data–evolution, challenges and research agenda," *International Journal of Information Management*, vol. 48, pp. 63–71, 2019.

[18] J. F. Sowa, Ed., *Principles of Semantic Networks - Explorations in the Representation of Knowledge*, ser. The Morgan Kaufmann Series in representation and reasoning. Morgan Kaufmann, 1991. [Online]. Available: https://doi.org/10.1016/C2013-0-08297-7

[19] E. M. Roth, D. D. Woods, and H. E. POPLE Jr, "Cognitive simulation as a tool for cognitive task analysis," *Ergonomics*, vol. 35, no. 10, pp. 1163–1198, 1992.

[20] E. Tom, A. Aurum, and R. T. Vidgen, "A consolidated understanding of technical debt," in *20th European Conference on Information Systems, ECIS 2012, Barcelona, Spain, June 10-13, 2012*, 2012, p. 16. [Online]. Available: http://aisel.aisnet.org/ecis2012/16

[21] A. Ampatzoglou, A. Ampatzoglou, A. Chatzigeorgiou, and P. Avgeriou, "The financial aspect of managing technical debt: A systematic literature review," *Information and Software Technology*, vol. 64, pp. 52–73, 2015.

[22] N. S. Alves, T. S. Mendes, M. G. de Mendonça, R. O. Spínola, F. Shull, and C. Seaman, "Identification and management of technical debt: A systematic mapping study," *Information and Software Technology*, vol. 70, pp. 100 – 121, 2016.

[23] J. Mostow, "Foreword what is ai? and what does it have to do with software engineering?" *IEEE Transactions on Software Engineering*, no. 11, pp. 1253–1256, 1985.

[24] D. Partridge, *Artificial intelligence: applications in the future of software engineering*. Halsted Press, 1986.

[25] L. Ford, "Artificial intelligence and software engineering: a tutorial introduction to their relationship," *Artificial intelligence review*, vol. 1, no. 4, pp. 255–273, 1987.

[26] A.-A. Tsintzira, E.-M. Arvanitou, A. Ampatzoglou, and A. Chatzigeorgiou, "Applying machine learning in technical debt management: Future opportunities and challenges," in *International Conference on the Quality of Information and Communications Technology*. Springer, 2020, pp. 53–67.

[27] M. I. Azeem, F. Palomba, L. Shi, and Q. Wang, "Machine learning techniques for code smell detection: A systematic literature review and meta-analysis," *Information and Software Technology*, vol. 108, pp. 115–138, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950584918302623

[28] W. E. Wong, N. Mittas, E. M. Arvanitou, and Y. Li, "A bibliometric assessment of software engineering themes, scholars and institutions (2013–2020)," *Journal of Systems and Software*, p. 111029, 2021.

[29] S. Wang, L. Huang, A. Gao, J. Ge, T. Zhang, H. Feng, I. Satyarth, M. Li, H. Zhang, and V. Ng, "Machine/deep learning for software engineering: A systematic literature review," *IEEE Transactions on Software Engineering*, 2022.

[30] R. Feldt, F. G. de Oliveira Neto, and R. Torkar, "Ways of applying artificial intelligence in software engineering," in *2018 IEEE/ACM 6th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering (RAISE)*. IEEE, 2018, pp. 35–41.

[31] P. Tonella, M. Torchiano, B. Du Bois, and T. Systä, "Empirical studies in reverse engineering: state of the art and future trends," *Empirical Software Engineering*, vol. 12, no. 5, pp. 551–571, 2007.

[32] M. Shaw, "Writing good software engineering research papers," in *25th International Conference on Software Engineering, 2003. Proceedings.* IEEE, 2003, pp. 726–736.

[33] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, 2014, pp. 1–10.

[34] B. Kitchenham and P. Brereton, "A systematic review of systematic review process research in software engineering," *Information and software technology*, vol. 55, no. 12, pp. 2049–2075, 2013.

[35] S. Fabbri, C. Silva, E. Hernandes, F. Octaviano, A. Di Thommazo, and A. Belgamo, "Improvements in the start tool to better support the systematic review process," in *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering*, 2016, pp. 1–5.

[36] N. B. Ali, K. Petersen, and C. Wohlin, "A systematic literature review on the industrial use of software process simulation," *Journal of Systems and Software*, vol. 97, pp. 65 – 85, 2014.

[37] *Supplementary Material. Managing Technical Debt Using Intelligent Techniques - A Systematic Literature Review*. Figshare, 7 2022. [Online]. Available: https://doi.org/10.6084/m9.figshare.21215192.v1

[38] T. B. Sheridan, "Computer control and human alienation," *Technology review*, vol. 83, no. 1, pp. 60–73, 1980.

[39] R. Feldt, "Do system test cases grow old?" in *2014 IEEE Seventh International Conference on Software Testing, Verification and Validation*. IEEE, 2014, pp. 343–352.

[40] P. Domingos, *The master algorithm: How the quest for the ultimate learning machine will remake our world*. Basic Books, 2015.

[41] A. Martini and J. Bosch, "The magnificent seven: towards a systematic estimation of technical debt interest," in *Proceedings of the XP2017 Scientific Workshops*, 2017, pp. 1–5.

[42] B. Pérez, "A semiautomatic approach to identify architectural technical debt from heterogeneous artifacts," in *European Conference on Software Architecture*. Springer, 2020, pp. 5–16.

[43] R. Marinescu, "Assessing technical debt by identifying design flaws in software systems," *IBM Journal of Research and Development*, vol. 56, no. 5, pp. 9:1–9:13, 2012.

[44] A. Martini, E. Sikander, and N. Medlani, "Estimating and quantifying the benefits of refactoring to improve a component modularity: A case study," in *2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, 2016, pp. 92–99.

[45] R. Plösch, J. Bräuer, M. Saft, and C. Körner, "Design debt prioritization: A design best practice-based approach," in *2018 IEEE/ACM International Conference on Technical Debt (TechDebt)*, 2018, pp. 95–104.

[46] H. Wang, M. Kessentini, W. Grosky, and H. Meddeb, "On the use of time series and search based software engineering for refactoring recommendation," in *Proceedings of the 7th International Conference on Management of computational and collective intElligence in Digital EcoSystems*, 2015, pp. 35–42.

[47] A. T. Misirli and A. B. Bener, "Bayesian networks for evidence-based decision-making in software engineering," *IEEE Transactions on Software Engineering*, vol. 40, no. 6, pp. 533–554, 2014.

[48] A. Tosun, A. B. Bener, and S. Akbarinasaji, "A systematic literature review on the applications of bayesian networks to predict software quality," *Software Quality Journal*, vol. 25, no. 1, pp. 273–305, 2017.

[49] M. V. Kosti, A. Ampatzoglou, A. Chatzigeorgiou, G. Pallas, I. Stamelos, and L. Angelis, "Technical debt principal assessment through structural metrics," in *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 2017, pp. 329–333.

[50] M. Ciolkowski, L. Guzmán, A. Trendowicz, and F. Salfner, "Lessons learned from the prodebt research project on planning technical debt strategically," in *International Conference on Product-Focused Software Process Improvement*. Springer, 2017, pp. 523–534.

[51] J. S. de Jesus and A. C. V. de Melo, "Technical debt and the software project characteristics. a repository-based exploratory analysis," in *2017 IEEE 19th Conference on Business Informatics (CBI)*, vol. 1. IEEE, 2017, pp. 444–453.

[52] M. Harman, S. A. Mansouri, and Y. Zhang, "Search-based software engineering: Trends, techniques and applications," *ACM Computing Surveys (CSUR)*, vol. 45, no. 1, pp. 1–61, 2012.

[53] R. M. Santos, M. C. R. Junior, and M. G. d. Mendonça Neto, "Self-admitted technical debt classification using lstm neural network," in *17th International Conference on Information Technology–New Generations (ITNG 2020)*. Springer, 2020, pp. 679–685.

[54] I. Sala, A. Tommasel, and F. Arcelli Fontana, "Debthunter: A machine learning-based approach for detecting self-admitted technical debt," in *Evaluation and Assessment in Software Engineering*, 2021, pp. 278–283.

[55] T. S. Mendes, D. A. Almeida, N. S. Alves, R. O. Spínola, R. Novais, and M. Mendonça, "Visminertd-an open source tool to support the monitoring of the technical debt evolution using software visualization," in *International Conference on Enterprise Information Systems*, vol. 2. SCITEPRESS, 2015, pp. 457–462.

[56] J. Flisar and V. Podgorelec, "Identification of self-admitted technical debt using enhanced feature selection based on word embedding," *IEEE Access*, vol. 7, pp. 106 475–106 494, 2019.

[57] X. Ren, Z. Xing, X. Xia, D. Lo, X. Wang, and J. Grundy, "Neural network-based detection of self-admitted technical debt: from performance to explainability," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 28, no. 3, pp. 1–45, 2019.

[58] B. Kitchenham, L. Pickard, and S. L. Pfleeger, "Case studies for method and tool evaluation," *IEEE software*, vol. 12, no. 4, pp. 52–62, 1995.

Danyllo Albuquerque received the MSc in informatics from Federal University of Paraiba, Brazil, in 2013. Currently, he is a PhD student in computer science in the Federal University of Campina Grande, Brazil. He is also a member of the Intelligent Software Engineering (ISE/Virtus) research group. His research interests are in the application of intelligent techniques to solve software engineering problems.

Everton Guimarães is an Assistant Professor at Penn State University. He has collaborated with many partners in industry and academia, both in Brazil and abroad, over the past 9 years. His most recent research investigates of problems related to software architecture and source code, and their impact on the overall software quality attributes (i.e. maintenance, evolution). He is also interested in research topics related to technical debt, architecture recovery techniques, pattern detection and mobile computing.

Graziela Tonin received the PhD degree in Computer Science from University of São Paulo (USP), Brazil, in 2018. She is assistant professor at the Federal University of Fronteira Sul. Her current research interests are in technical Debt, Agile Methods, metrics, experimental Software Engineering and software evolution.

Pilar Rodríguezs is an Assistant Professor at Universidad Politécnica de Madrid (Spain). She received her PhD in Computer Science (2013) from University of Oulu (Finland). Dr. Rodríguez's research interests focus on software engineering and empirical software engineering including software development processes, with area of interest in value-based decision-making in software development, Lean and Agile methods, and software process improvement.

Mirko Perkusich received his PhD degree in Computer Science in 2018. He is a Research Manager at VIRTUS, leading the Intelligent Software Engineering (ISE/VIRTUS) research group. His current research interests are in applying intelligent techniques, including recommendation systems and Bayesian networks, to solve complex software engineering problems.

Hyggo Almeida is a professor at the Computer and Systems Department, Federal University of Campina Grande (UFCG) since 2006. Dr. Almeida has got his Ph.D. in Electrical Engineering and M.Sc. in Computer Science both from the Federal University of Campina Grande in 2007 and 2004, respectively. He is currently the head of Intelligent Software Engineering group, and Founder and Director of Operations at VIRTUS Innovation Center (VIRTUS/UFCG). He is a researcher at Embedded Systems and Pervasive Computing Laboratory (Embedded/UFCG). He is also Executive Director of EMBRAPII Unit at CEEI-UFCG, with more than 150 RD&I projects developed in cooperation with industrial companies within the area of Information, Communication and Automation Technologies. His current research interest is applying intelligent techniques to solve complex software engineering problems.

Angelo Perkusich (Member IEEE) is a professor at the Electrical Engineering Department, Federal University of Campina Grande (UFCG) since 2002. Dr. Perkusich got his Ph.D. and Master's degrees in Electrical Engineering both from the Federal University of Paraiba in 1987 and 1994, respectively. He was a visiting researcher at the Department of Computer Science, University of Pittsburgh, PA, USA, from 1992 to 1993. He is currently the principal investigator of research projects financed by public institutions such as FINEP (Brazilian Agency for Research and Studies) and CNPq (Brazilian National Research Council), as well as private companies. He is the founder and Director of VIRTUS Innovation Center (VIRTUS/UFCG) and Embedded and Pervasive Computing Laboratory (Embedded/UFCG). He has over 30 years of teaching experience in the university as well as training courses for industry in the context of software for real-time systems, software engineering, embedded systems, computer networks, mobile pervasive computing, and formal methods.

Ferdinandy Chagas received the MSc degree in computer science from Federal University from Semi-Arid Region, Brazil, in 2013. Currently, he is a PhD student in computer science in the Federal University of Campina Grande, Brazil. He is also an assistant professor in the Federal University of the Semi-Arid Region. His research focuses on multiagent systems, ontologies, software process improvement models. He is a member of the Intelligent Software Engineering Group (ISE/Virtus).