IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 9 November 2024, accepted 31 December 2024, date of publication 10 January 2025, date of current version 15 January 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3528297

SURVEY

SOA Services Identification and Design Methods From Business Models: A Systematic Literature Review

REDOUANE BLAL[1], ABDERRAHMANE LESHOB[1], HAFEDH MILI[1], IMEN BENZARTI[2], PIERRE HADAYA[1], AND RAQEEBIR RAB[3]

[1]Département d'Analytique, Opérations et Technologies de l'Information, Université du Québec à Montréal, Montreal, QC H3C 3P8, Canada
[2]Département de Génie Logiciel et des TI, École de Technologie Supérieure de Montréal, Montreal, QC H3C 1K3, Canada
[3]Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka 1208, Bangladesh

Corresponding author: Redouane Blal (blal.redouane@courrier.uqam.ca)

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

ABSTRACT Modern organizations are process-oriented. To remain competitive in the digital transformation era, these organizations design and implement Information Systems (IS) to support their business processes. The emergence of the Service-Oriented Architecture (SOA) style made it possible to design IS architectural models that meet software quality criteria and are aligned with the organizations' business models. However, designing such SOA services is a complex task that requires extensive knowledge and skills in software architecture and the business domain. Researchers and practitioners have proposed several methods to derive SOA services from business models during the last two decades. However, SOA design initiatives from business models still fail. Existing methods and processes to design process-aware IS have limitations related to their usability and implementation complexity. These limitations triggered the necessity for a survey that extracts more information about how to design SOA architectural models from business models. This work proposes a systematic literature review (SLR) to establish the state of the knowledge about the existing methods that derive service models from business models. This SLR provides practitioners, such as solutions architects, business architects, and application architects, a comprehensive overview of available methods to derive SOA architectural models from business models, helping them design SOA services and build effective software solutions. We selected forty-one primary studies published between 2006 and 2023. We compared selected methods according to seven specific criteria, namely: design life cycle coverage, detailed specification support, SoaML support, service granularity, the use of patterns, automation, and tool support. The results confirm that SOA is an established architectural style for building effective ISs that support organizations' business processes. As far as we observed from comparing selected methods according to the selected criteria, our findings explain the need for a new method that provides organizations with an easy-to-use and comprehensive process to derive quality SOA models from business models.

INDEX TERMS Systematic review, service-oriented architecture, SOA, business models, service design, model transformation.

I. INTRODUCTION

Nowadays, complexity characterizes the environment in which organizations evolve and collaborate. To remain competitive, organizations must stay abreast of changes and continually modernize through the design and development of information systems (IS) to support their business processes [1]. However, IS development is a complex and laborious task that requires resources and specific skills [2].

The advent and rapid adoption of the Service-Oriented Architecture (SOA) paradigm represent a shift in IS design and development [3]. SOA introduced a modular

The associate editor coordinating the review of this manuscript and approving it for publication was Engang Tian.

design approach that simplifies the IS design task. SOA emphasizes reuse, flexibility, scalability, and interoperability, allowing organizations to integrate heterogeneous systems and adapt to evolving business needs more effectively [4]. SOA principles remain the foundation of current and future trends for designing and developing service-based systems [5].

Despite the SOA style's multiple benefits, organizations face many challenges when implementing SOA service-based solutions, considering the complexity of the SOA service design process [6]. Over the last decades, scholars and practitioners alike proposed several service design methods to overcome these challenges. However, the proposed methods still show limitations regarding their usability and implementation complexity.

A literature review describes, understands, explains, or tests theories [7]. Over time, several literature reviews were conducted to portray the state of the knowledge and to reveal the gaps in SOA service identification and design methods [8], [9], [10], [11], [12], [13], [14], [15], [16]. Existing reviews covered previously published studies regardless of the starting point of the method to derive SOA services. In this work, we conduct a Systematic Literature Review (SLR) to identify and describe the available methods to design services according to the SOA style using business models as input. Moreover, while existing reviews used diverse comparison criteria [6], this study introduces new critical quality criteria to compare the available methods. Thus, it compares the methods' coverage of SOA service design stages, level of automation, tool support, use of best practices such as patterns, support for service granularity, and SoaML support. Our selected comparison criteria are described in more detail in subsection VII-B.

This work aims to identify, evaluate, and interpret available studies about the research topic. Thus, according to the guidelines from [17], this SLR process is organized into three main phases: i) Planning the review, ii) Conducting the review, and iii) Reporting the review.

The remainder of this paper is structured as follows. Section II explains the background underneath the SOA identification and specification methods. Section III surveys the related work and explains the motivation for performing this SLR. Section IV presents the SLR methodology. In section V, we describe the review protocol and identify the research questions, the search strategy, and the relevance criteria. Section VI presents the details of the process to conduct this review. In section VII, we analyze and report the results from the SLR. Then, we briefly discuss our findings about existing methods. Section VIII discusses the threats to the validity of the SLR. We conclude in section IX.

II. BACKGROUND

This section introduces the SOA architectural style and its benefits. Then, it presents definitions and types of business models to explain how various input models could be used to derive SOA services.

A. SOA

SOA is a paradigm that gained prominence in the late 1990s and early 2000s as a response to the need for reusable, flexible and scalable software solutions that could be easily integrated into different applications [4]. SOA has evolved from previous architectural paradigms and development approaches, including Object-Oriented Development (OOD), Component-Based Architecture (CBA), and Distributed Systems [18], [19].

OOD introduced the concepts of objects and interfaces. Objects encapsulate data and behaviour. Interfaces define the contract to interact with the objects by specifying how their methods can be invoked [19]. OOD paradigms focus on packaging data with operations. An object exposes a structure but cannot express semantics, which can only be expressed through comments in the class definition [20].

CBA is based on assembling pre-built and reusable software components. According to [19], CBA is a development paradigm that extends OOD with the notion of "Component". The latter is a larger-grained unit than objects.

According to [21], Distributed Systems are a collection of autonomous computing elements that appear to their users as a single coherent system. This implies interconnected components communicating remotely over a network to ensure they appear as a single system.

According to [22], SOA is a common approach to designing and developing Process-Aware Information Systems (PAIS).[1] SOA established an architectural style to enhance the enterprise's efficiency by positioning services as the primary means for supporting the business [23]. SOA has emerged as a paradigm for packaging a set of capabilities as services [24]. Services are used to build and provide software solutions to solve business problems. Hence, SOA-based systems are decomposed into loosely coupled, interoperable services that encapsulate specific business functionalities. Authors in [25] define the SOA service concept as a self-contained business functionality with well-defined and discoverable interfaces. SOA services expose well-defined interfaces that can be discovered, invoked, and composed dynamically [3].

Moreover, SOA introduced a set of core principles and service quality attributes such as cohesion, coupling, and granularity. According to [23], the SOA principles of service design are:

- Standardized Service Contract: Services within the same service inventory comply with the same contract design standards.
- Service Loose Coupling: Service contracts impose low consumer coupling requirements and are themselves decoupled from their surrounding environment.
- Service Abstraction: Service contracts only contain essential information, and information about services is limited to what is published in service contracts.

[1]Process-Aware Information System (PAIS) refers to systems built to support the organization's business processes [22].

- Service Reusability: Services contain and express agnostic logic and can be positioned as reusable enterprise resources.
- Service Autonomy: Services exercise a high level of control over their underlying runtime execution environment.
- Service Statelessness: Services minimize resource consumption by deferring state information management when necessary.
- Service Discoverability: Services are supplemented with communicative metadata that can be effectively discovered and interpreted.
- Service Composability: Services are effective composition participants, regardless of the size and complexity of the composition.

SOA is a technology-independent architectural style. Early SOA implementations relied on the Enterprise Service Bus (ESB) to facilitate communication between services, orchestrating interactions and handling tasks like security, message transformation, and routing [4]. However, the ESBs are complex and heavy. Their centralized nature sometimes leads to performance bottlenecks. Around the mid-2000s, the REST (Representational State Transfer) architectural style began to gain popularity as an alternative [26]. REST advocates focusing on simplicity and performance using simple, stateless HTTP communication and lightweight formats such as JavaScript Object Notation (JSON).

Since the beginning of the 2010s, Microservices have evolved as a more granular and modern approach to addressing many traditional SOA services design challenges, such as granularity [27]. Microservices encourage decentralized governance and data management, with each service having its own database. Microservices typically communicate via lightweight protocols like REST [28].

SOA principles such as loose coupling, composability, and reuse laid the foundation for novel service-based architectures. However, new approaches like Microservices have refined and simplified these foundations, making them more scalable, flexible, and aligned with today's modern software development practices, such as DevOps and Continuous Integration/Continuous Deployment (CI/CD).

B. BUSINESS MODELS

SOA emphasizes using flexible, reusable, and interoperable services to enhance software development and system integration that fulfill business processes and goals. Identifying and specifying the appropriate services is essential for building a service-based solution [29]. According to [23], there are three possible strategies to derive SOA services: top-down, bottom-up, or meet-in-the-middle. Bottom-up and meet-in-the-middle strategies are useful when SOA-based systems are derived from existing systems [30]. In a top-down strategy, services are derived from business models before they undergo technical specification [9]. According to [31], using a top-down strategy to derive SOA services models improves their alignment with the business models. *This work focuses on proposed methods that use a top-down strategy to identify SOA services from business models.*

Business models are representations that outline real-world business activities independently of how the systems are implemented [32]. Various business models exist, including Business Process Models (PM), Value Models (VM), and Goal Models (GM), each serving distinct purposes to describe various aspects of the business [33]. Together, the business models provide a comprehensive view of organizational strategy, operations, and value.

PMs describe the ''how'' of achieving specific goals by mapping out the sequence of organizational activities, tasks, and decisions [33]. They focus on operational efficiency and effectiveness, capturing workflows and dependencies within the processes [34]. PMs are used for operational analysis, ensuring that activities align with organizational goals. PMs can be created using various modeling techniques and notations, including Business Process Model and Notation (BPMN) [35], Unified Modeling Language (UML) [36], and Event-driven Process Chains (EPC) [37].

VMs, in turn, capture the value creation from the performance of business activities [33]. They illustrate ''what value'' an organization provides and how value is created, exchanged, and delivered among business actors [38], [39]. VMs are used in business design and strategic analysis, identifying value creation, offerings, and exchanges. VMs can be obtained using a business ontology, such as e-BMO [40], REA [38], [39], and e3-value [41].

Finally, GMs capture the goals and objectives that an organization seeks to achieve, how these objectives relate to one another, and how they are connected to the deployed means and actions (e.g., services). GMs clarify the rationale and motivation for answering the 'why' questions of the business [30], [33]. They help align resources with organizational priorities and identify potential conflicts or dependencies between goals. Several goal-oriented modeling approaches exist. For example, Goal-oriented Requirement Language (GRL) [42], Knowledge Acquisition in Automated Specification (KAOS) [43], and I* [44].

III. RELATED WORK

An SLR helps define each research project's context and objectives to demystify its complexity [45]. Before starting this work, we reviewed existing literature on service identification and design methods. Specifically, we examined the scope of previous reviews and their criteria for comparison. This section provides a summary of relevant contributions from the previous reviews and the motivation for this work.

In [10], the authors conducted a criteria-based literature review to compare existing approaches using six groups of characteristics. Their work used comparison criteria such as input type, service identification strategy (i.e., top-down, bottom-up, or meet-in-the-middle), service design life cycle

coverage, tool support, value creation, and service granularity. According to the authors in [10], existing approaches lack the business perspective. Hence, they considered the latter a precondition for successful and efficient SOA-based solution implementation. The study reveals that each method has strengths and weaknesses. It concludes that combining various criteria from multiple methods may offer a more effective solution and build an adaptive method.

Gu and Lago conducted an SLR about existing service design methods selected from 30 studies [11]. They concluded that existing methods differ in basic aspects, especially input and output types, techniques, and strategies. The study compared the methods based on efficiency, effectiveness, and adaptability. It concluded that no single method is universally applicable, as each has unique advantages and limitations. Based on the study findings, the authors proposed a comparison framework to help practitioners assess which suitable method fits their needs.

In turn, the authors in [12] conducted an SLR specifically tailored for Software Product Lines (SPLs). They compared existing service design methods for SPLs' using a combined set of criteria introduced previously in [10] and [11]. Selected criteria include service granularity, input models, output artifacts, and used techniques. The paper outlined that existing methods need adjustments to suit the unique requirements of SPLs'. More precisely, the method must manage variability and feature modularity, which are fundamental to SPLs.

In [13], the authors assessed existing service identification methods using specific comparison criteria such as service design life cycle coverage, the perspective level of the guidelines, reuse of existing techniques, clustering, and the use of service typology/classification. The authors in [46] surveyed service identification approaches. They assessed the strengths and weaknesses of existing approaches according to their characteristics. These characteristics include the level of formalization, overall design process coverage, applied techniques, the perspective level of the analysis, the use of an optimization approach, and the availability of supporting tools. In [16], the authors compared existing methods in terms of their basic characteristics, such as the design starting point, modeling covered phases, and used techniques. Findings from [13], [16], and [46] provided meaningful insights to help with future research directions. However, they covered fewer and older versions of existing SOA service identification methods.

In [8], the authors examined existing literature to identify high-value design activities shared by various service identification and design methods. They highlighted model-to-model transformation, business model decomposition, value analysis, pattern matching, and ontology mapping. Value analysis explores business value change within the enterprise value chain. Pattern matching identifies services from business models by matching existing patterns. Ontology mapping helps to identify services from business models based on a set of ontology definitions [8]. The study performed a comparative analysis using efficiency, scalability, alignment with business objectives, and adaptability criteria. The authors concluded that future research needs to overcome shared challenges across existing methods, including the difficulty of maintaining flexibility for future changes and the challenge of accurately aligning with evolving business processes. Moreover, they highlighted the importance of tools and techniques to support dynamic service adaptation as organizations grow.

Authors in [9] conducted a detailed investigation of existing service design methods and compared them using a set of relevant criteria. The criteria were inferred from various perspectives of the OASIS reference architecture for SOA [47]. Their evaluation framework included criteria such as business alignment, flexibility, granularity, reusability, and complexity. The study provided a structured overview of existing methods and classifications based on each method's foundational principles. It revealed that the selected methods differ mainly in terms of the input model, level of automation, and techniques used. Additionally, the authors highlighted how some methods are better suited to align with business goals while others excel in technical modularity or scalability.

In [14], the authors analyzed and discussed the automation issues of existing service design methods by examining their techniques and the available tools that support automation. They concluded that when implemented, automation techniques vary depending on the service design phase. The study outlined the correlation between the lack of automation and poor alignment between service and business models.

In [15], Fausel and Hussein presented another evaluation of existing service design methods based on selected criteria, such as SOA design strategies and coverage, the nature of instructions, the use of service hierarchy and types, output type, and tool support. For a more precise comparison, the authors suggested selecting criteria based on the scope of the existing methods.

In [48], authors identified commonly used comparison criteria from previous reviews. They identified sixteen (16) criteria, namely: service design life cycle coverage, approach/strategy, input artifact, used techniques, types of services, service descriptions/output, service quality attributes, service granularity, comprehensiveness, systematic (i.e., detailed guidance), availability, tool support, adoption of existing practices, validation, configurability (or adaptability), and applicable domain.

In [6], the authors conducted a literature review to explore the existing challenges of SOA design methods. The review findings summarize the top challenges, notably the service granularity principle, systematic approach and automation, the comprehensiveness of the coverage, tool support, input artifact, and configurable solution (i.e., flexible adaptation of methods).

Previous surveys concluded that no single service identification method fully meets all criteria. They suggested that organizations may benefit from hybrid methods integrating

elements from multiple approaches. Authors in [9] suggested that future empirical studies validate the effectiveness of these methods in real-world scenarios. However, none of the subsequent reviews evaluated the implementation of earlier recommendations in more recent proposed methods. Findings from existing surveys provide meaningful insights for the research. Previous reviews covered available SOA service identification methods regardless of their input and service identification strategies. Moreover, earlier studies used various comparison criteria based on their scope.

This work conducts an SLR about existing methods to derive SOA service design models from business models. In addition to including recently proposed methods, this work differentiates itself from previous reviews in two ways. First, it focuses on methods for SOA service design using business models as input. Second, it includes an analysis focusing on the most critical quality criteria to compare existing methods. It examines SOA life-cycle coverage (i.e., identification, specification, and realization). It verifies whether the method supports the service granularity principle and assesses its automation level and tool support. Moreover, it uses new criteria such as detailed specification support, support of best design practices such as patterns, and SoaML support for service modeling and specification. Table 9 shows the list of adapted comparison criteria.

IV. LITERATURE REVIEW METHODOLOGY

According to [45], an SLR uses a well-defined methodology to identify, analyze, and interpret all available evidence related to a specific research question in an unbiased and repeatable way. A systematic review aims to present a fair appreciation of the research subject using a trustworthy and rigorous methodology [17]. The required methodology consists of specific activities and guidelines to achieve the expected objective [49]. This work adheres to the guidelines on conducting a systematic literature review provided by the work in [17]. Therefore, we organized this SLR's activities into the following three main phases: 1) Planning, 2) Conducting, and 3) Reporting.

During the planning phase, we define the research questions to be addressed, the search strategy to identify primary studies (PSs), and the inclusion and exclusion criteria. Before conducting this review, we assessed its pertinence by comparing its aimed contributions to existing literature reviews. The conducting phase consists of performing planned activities according to the defined strategies. Hence, the first activity during this phase is to search for and identify relevant PSs. A study is considered relevant if it proposes a method for identifying and specifying SOA services from business models. Then, we extract, retrieve, and summarize relevant data from each selected PS to answer the research questions.

Lastly, during the reporting phase, we analyze each method using a defined comparison criteria framework. The criteria were justified according to identified key characteristics of the SOA service design methods (see section VII-B). Then, we interpret, report, and summarize the review search results. Analysis activities synthesize collected data from the review to answer research questions. Table 1 describes the activities and phases of the methodology designed to conduct this review.

TABLE 1. Key phases and activities of this review process.

Phase	Activities
Planning	Identify the research questions Identify the search strategy
Conducting	Select relevant studies Collect and chart the data
Reporting	Summarize and report the results

V. PLANNING THE REVIEW

In the preceding sections, we defined this review's motivation and methodology. Before undertaking this work, we verified its relevance given previous reviews' scope and comparison criteria (see section III). According to [17], the planning phase defines the review protocol. This protocol consists of identifying the research questions, describing the search strategy, and determining the inclusion and exclusion criteria and data extraction method.

A. IDENTIFYING THE RESEARCH QUESTIONS

The main goal of this work is to identify and examine the characteristics of existing design methods that derive SOA services from business models. This section identifies relevant research questions related to the SLR's research subject. Table 2 shows the research questions and the motivation behind them.

B. SEARCH STRATEGY

We defined our search strategy to ensure an exhaustive search with comprehensive results. To achieve this goal, we considered a selection of keywords and their synonyms. Given this study's context and research questions, we identified the concepts of ''Service'', ''Identification'', ''Method'', and ''Service-Oriented Architecture''.

From this list of keywords, we searched for related synonyms using an online synonym tool.[2] Once a list of synonyms was built, we examined the pertinence of each available synonym based on the context of this study to select an optimal set of additional keywords, including: ''derivation'', ''deriving'', ''design'', ''approach'', and ''methodology''. We also used the ''SOA'' acronym for ''Service-oriented architecture''. Additionally, we added keywords such as ''business model'', ''process model'', ''goal model'', and ''value model''. These keywords apply exclusion criteria regarding the input types used by the methods to limit the search results to the most relevant studies. The combination of the elected keywords resulted in the following generic search string (SQ):

[2]https://www.thesaurus.com

TABLE 2. Research questions.

Research question (RQ)	Motivation
RQ1: What are the SOA service design phases covered by existing methods?	Compare the methods according to the extent of SOA design phase coverage, namely service identification, specification, and realization phases.
RQ2: What are the methods that support detailed service specification?	Discover the methods that support the detailed specification and the techniques used.
RQ3: What are the methods that support SoaML in describing the services?	Compare the methods based on their adoption of the SoaML language to specify the SOA models.
RQ4: What are the methods that use formalized solutions such as patterns to identify and/or specify the services?	Classify the methods based on their usage of formalized best practices and patterns.
RQ5: What are the methods that consider the service granularity principle?	Identify the methods that consider the granularity of the services and their supported granularity levels.
RQ6: What are the levels of automation among the available methods?	Compare and classify the method according to the level of automation.
RQ7: What are the methods that offer supporting tools?	Identify the methods that propose tools to perform the design activities

TABLE 3. Selected databases.

Database name	URL
ACM Digital Library	http://dl.acm.org
Scopus	http://www.scopus.com
IEEEXplore	http://ieeexplore.ieee.org
DBLP Computer Science Bibliography	http://www.dblp.org
SpringerLink	https://link.springer.com

("service identification" OR "deriving service" OR "service derivation" OR "service design" OR "service specification") AND ("business process model" OR "goal model" OR "value model" OR "business model") AND (method OR methodology OR approach) AND ("service-oriented architecture" OR "service oriented architecture" OR soa).

To retrieve an exhaustive list of relevant studies, we carefully selected five online scientific databases based on the following criteria: (i) relevance and coverage of the research field, (ii) frequency of updates, (iii) availability, and (iv) quality of the obtained search results. Table 3 shows the selected databases.

C. INCLUSION AND EXCLUSION CRITERIA

Identified keywords are used in various contexts and fields. Hence, considering the potential number of research works that could be related to these keywords, we refined the search results using inclusion and exclusion criteria to limit the selection to studies that propose a method for deriving services from business models. Thus, we defined inclusion and exclusion criteria based on the research questions described in Table 2. The justifications for the exclusion criteria are presented in Table 4.

Exclusion criteria:

- Papers presenting a prior version of the same method.
- Papers that do not propose a method or an approach.
- Papers where the method/approach input is not a business model.
- Papers that are not accessible.
- Publication before 2006.
- Papers not written in English.

Inclusion criteria:

- Papers that propose methods using business models as inputs.
- Published between 2006 and 2023.
- Peer-reviewed.
- English writing studies.

D. DATA EXTRACTION STRATEGY

According to [17], a data extraction method defines how the information required from each paper will be extracted. Data extraction is required to analyze the articles, examine the key characteristics of the methods, and answer the research questions. During this stage, we collect information about the selected papers, including the title, abstract, author names, publishing year, paper type, and journal. From the full text of each selected article, we collect information based on the following steps:

1) The input and output models.
2) Design stages covered by the method (i.e., identification, specification, and realization).
3) Techniques used at every stage covered by the method.
4) If the method supports detailed service specification models (Yes/No).
5) If the method covers service granularity (Yes/No).
6) If the method uses a pattern-based approach to identify and/or design the services (Yes/No).
7) Determine the method's automation level.
8) If the method is supported by a tool (Yes/No).

Table 5 shows the details of collected data description and sources.

VI. CONDUCTING THE REVIEW

This phase involves two main activities with multiple steps. The first involves searching for primary studies and selecting relevant articles. The second consists in collecting and recording relevant data from selected articles according to an explicitly established protocol.

A. SELECTING RELEVANT STUDIES

We performed a five-stage search and filtering activity to select articles related to our research subject. Figure 1 shows

TABLE 4. Exclusion criteria and justification.

Exclusion criteria	Justification
Papers presenting a prior version of the same method.	Avoid redundant data from improved versions of the same methods.
Papers that do not propose a method or an approach.	Our work objective is to compare existing design methods; we ignore documents that don't have a structured approach for service identification or specification.
Papers where the method/approach input is not a business model.	The scope of this SLR is methods that derive services from business models.
Papers that are not accessible.	We ignored papers that were not publicly accessible since we could not access their data.
Papers not written in English.	Researchers can read, interpret, and accurately summarize papers written in English.
Publication before 2006.	We established a time scope to limit the publication period coverage of this SLR between 2006 and 2023.

TABLE 5. Data extraction protocol.

Collected Data	Description	Source
Authors	Names of authors	Paper reference
Title	Article title	Paper reference
Year	Publishing year	Paper reference
Paper Type	Conference or Journal paper or book section	Paper reference
Journal/ Conference/ Book name	The name of the conference, the journal, or the book.	Paper reference
Abstract	Abstract section	Paper reference
Input Model	What is the input models' type and representation?	Paper full text
Output Model	How are identified services described?	Paper full text
Design Phase Coverage	What service design phases are covered by the method?	Paper full text
Identification Techniques	What is the approach/technique used to identify the services?	Paper full text
Specification Techniques	What is the approach/technique used to generate high-level service models?	Paper full text
Detailed specification approach	What is the approach/technique used to generate detailed service specification models?	Paper full text
Service Granularity	What approach is used to define an appropriate service granularity level?	Paper full text
Pattern-based	Does the method use a Pattern-based technique/approach?	Paper full text
Automation level	What techniques are used to automate the method's activities?	Paper full text
Tools	What tools are available to support the performance of the method?	Paper full text

the selection and filtering process performed to identify relevant PS. In the following, we describe the different stages:

- **Stage 1**. For each database identified in Table 3, we customized a search query using a combination of selected keywords linked with logical connectors (e.g., AND, OR). Then, we applied the required modifications to ensure compatibility with each search engine's specifics.
  We used additional filters to obtain precise results from the queries when the database graphic interface allowed it. Hence, on the ACM Digital Library Database, we limited the search to the ''Management Information Systems'' and ''Computer Science'' fields. Likewise, on the SpringerLink database, we limited the search to the disciplines of ''Computer science'' or ''Business and management''), and to sub-disciplines of ''Information systems applications'', ''Software engineering'', ''IT in Business'' or ''Management of computing and information systems''. Finally, we applied a filter for the search on the IEEEXplore database, limiting the results to papers from conferences, journals, and magazines. Furthermore, we limited our search queries to peer-reviewed English studies published between 2006 and 2023. Table 6 shows used queries for each database. The combined results from the search queries performed on each database returned 771 reference documents. We recorded the selection using the Mendeley reference management tool ([50]). Table 6 depicts the number of references returned by each database query.
- **Stage 2.** To expand the coverage of this review, we identified 11 literature review articles from selected references. We applied forward and backward snowballing to minimize the threats of missing essential studies due to query string terms [51]. Forward snowballing refers to using primary study references to identify new papers. Backward snowballing refers to identifying new papers citing the paper being considered. We iterated from backward to forward snowballing and stopped the iteration process when we found no new candidate paper. From the snowballing search, we added another 121 references. At this stage, we selected a total of 892 articles.
- **Stage 3.** Using the reference management tool, we removed 88 duplicates and 11 literature review articles to retain 793 references for further consideration.
- **Stage 4.** At this stage, we searched for the original document of each selected reference in the previous step. Hence, we ignored articles that were not publicly accessible. Then, we performed a preliminary screening by consulting each article's title and abstract from

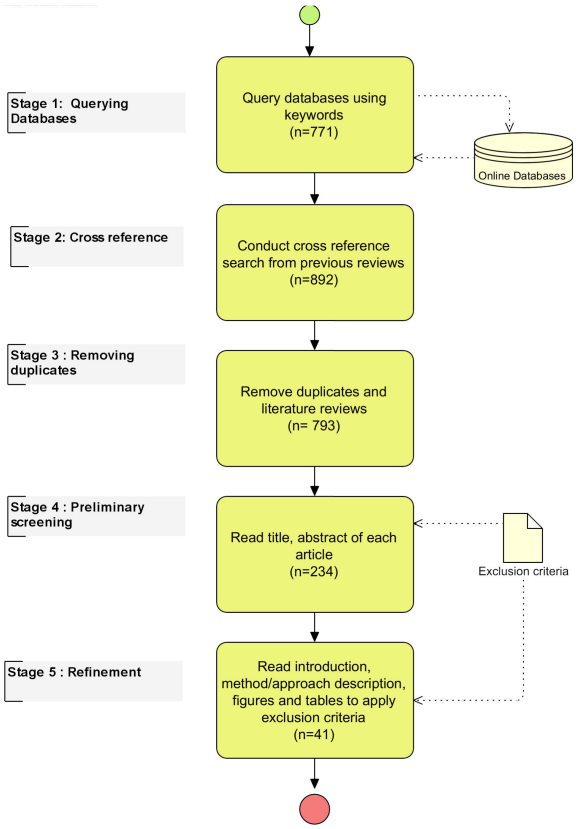

FIGURE 1. Articles selection and filtering process.

the retained selection. This task was accomplished and reviewed by two or more researchers. Hence, we removed 559 articles that met one or more of the identified exclusion criteria. Thus, we retained 234 documents for a more in-depth examination.

- **Stage 5.** We examined the remaining selection more deeply and consulted each of the 234 articles' introduction, method (or approach) descriptions, figures, and tables. First, we verified whether the article proposes a service identification or design method. Then, we identified the method's input and removed articles where the input is not a business model as defined above (see subsection V-A). Each article was read and examined as many times as needed by the researchers to ensure that all potentially relevant papers were selected.

  Furthermore, to refine the selection, we compared the details of methods from the same authors and kept only the latest version. At this stage, we eliminated 193 articles from the list and retained 41 articles for further analysis. Figure 1 shows the details of the selection process.

B. COLLECTING AND CHARTING THE DATA

During this step, we read and analyzed the 41 selected articles and extracted the most relevant data about the examined methods. Thus, we systematically performed the data extraction activities described in the subsection V-D, and Table 5. The collected data was concealed into a spreadsheet for processing and analysis.

VII. REPORTING THE RESULTS

This section presents an analysis of collected data from the review process about the available methods. First, we present information related to retained PSs. Second, we define a framework to compare selected methods based on different criteria. Then, we analyze their key characteristics. Finally, we synthesize the results to answer the research questions presented in Table 2.

A. PUBLICATION-RELATED DATA

During this SLR, we selected forty-one (41) studies that propose a method that derives SOA services from business models.

1) PUBLICATION DATE

As shown in Table 7, we found that 24 articles (more than 60 %) from this SLR were published over the first six years, from 2007 to 2013. The novelty and trending concepts, due to the shift in paradigms in designing software solutions, could explain the higher number of publications during that period. Publications were evenly distributed over the subsequent years. In recent years (i.e., between 2019 and 2023), we found 9 (21.95 %) new publications. Figure 2 shows the annual distribution of the collected articles.

2) PUBLICATION VENUES

To better understand the origin of publications, we have divided them into journals, book series, and conference proceedings. Among the selected studies, 18 (46%) are journal articles, 15 (39%) are conference papers, and 6 (15%) are book sections. Figure 3 shows the paper type distribution. The Table 8 shows the list of papers, journals, and books.

B. COMPARISON FRAMEWORK

We propose a seven-criteria comparison framework to analyze, classify, and summarize the collected data from this SLR. Criteria were defined and customized according to specific characteristics that impact either the use of the method or the quality of the derived services. Seven criteria are defined as follows:

1) **SOA service design life cycle coverage:** This criterion refers to whether the method covers all or fewer stages of the SOA service design life cycle. According to [31], the SOA service design process includes three main stages: identification, specification, and realization.[3]
   - The identification stage is the key to the service design methods. It involves activities to determine

[3]These stages depend on each other results. Thus, the method coverage might consist of one stage, two, or all of the three stages.

TABLE 6. Customized search queries.

Database name	Customized Search Query	Results
ACM Digital Library	[[Full Text: "service identification"] OR [Full Text: "deriving service"] OR [Full Text: "service derivation"] OR [Full Text: "service design"] OR [Full Text: " service specification"]] AND [[Full Text: "business process model"] OR [Full Text: "goal model"] OR [Full Text: "value model"] OR [Full Text: "business model"]] AND [[Full Text: method] OR [Full Text: methodology] OR [Full Text: approach]] AND [[Full Text: "service-oriented architecture"] OR [Full Text: "service oriented architecture"] OR [Full Text: soa]] AND [E-Publication Date: (01/01/2006 TO 31/12/2023)]	73
Scopus	(TITLE-ABS-KEY ("service identification" OR "deriving service" OR "service derivation" OR "service design" OR "service specification") AND ALL ("business process model" OR "goal model" OR "value model" OR "business model") AND ALL (method OR methodology OR approach) AND TITLE-ABS-KEY ("service-oriented architecture" OR "service oriented architecture" OR soa)) AND (LIMIT-TO (PUBSTAGE , "final")) AND (LIMIT-TO (SUBJAREA , "COMP") OR LIMIT-TO (SUBJAREA , "BUSI")) AND (LIMIT-TO (LANGUAGE , "English")) AND (LIMIT-TO (DOCTYPE , "cp") OR LIMIT-TO (DOCTYPE , "ar") OR LIMIT-TO (DOCTYPE , "ch"))	95
IEEEXplore	("Full Text Only":"service identification" OR "Full Text Only":"deriving service" OR "Full Text Only":"service derivation" OR "Full Text Only":"service design" OR "Full Text Only":"service specification") AND ("Full Text Only":"business process model" OR "Full Text Only":"goal model" OR "Full Text Only":"value model" OR "Full Text Only":"business model") AND ("Full Text Only":method OR "Full Text Only":methodology OR "Full Text Only":approach) AND ("Full Text Only":"service-oriented architecture" OR "Full Text Only":"service oriented architecture" OR "Full Text Only":soa)	119
DBLP Computer Science Bibliography	identif|deriv|design|speci service method|approach "service-oriented architecture"|SOA (*)	25
SpringerLink	("service identification" OR "deriving service" OR "service derivation" OR "service design" OR "service specification") AND ("Method" OR "Methodology" OR "Approach") AND (("Service Oriented Architecture" OR "Service-oriented Architecture" OR SOA) AND ("Business process model" OR "goal model" OR "value model" OR "business model")) AND (Date published : (01/01/2006 TO 31/12/2023)) And Languages : (English) Limited to Disciplines ("Computer science" OR "Business and management") AND Subdisciplines ("Information systems applications" OR "Software engineering" OR "It in business" OR "Management of computing and information systems") AND (Content-type : (Article OR "Conference proceedings" OR Conference paper)	459
(*) We used "Business process model, goal model, value model, and business model" as criteria to refine the query results in the subsequent phases of the process.		

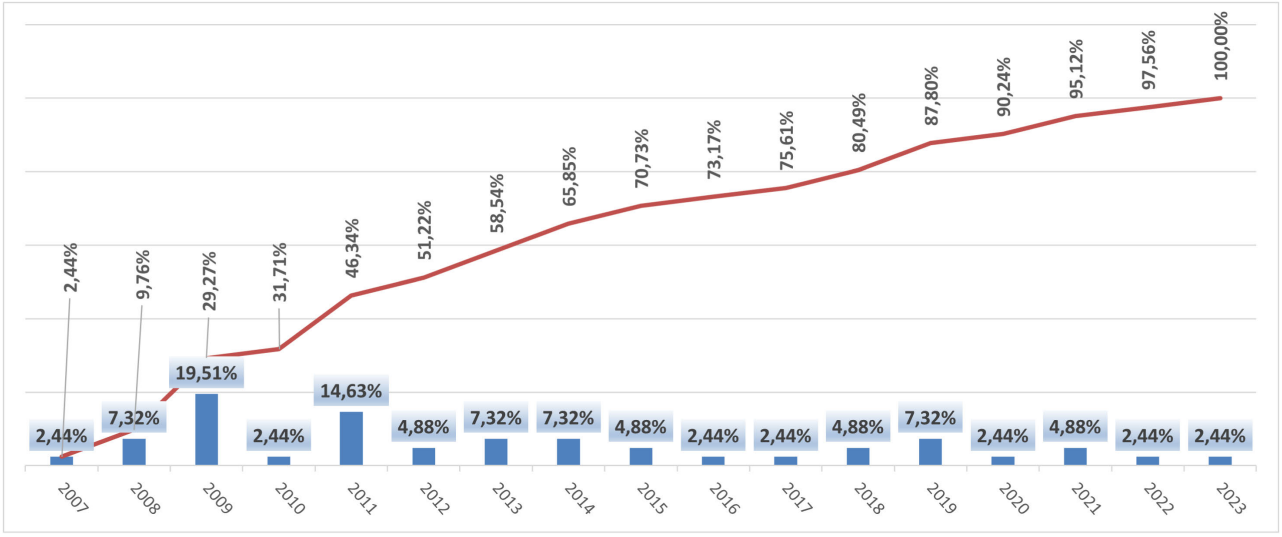

FIGURE 2. Selected articles per year.

the list of candidate services. Nevertheless, service descriptions can start within identification phases [9].

- The specification stage is the core of the service modeling activity. Service specification involves defining high-level service models [29], [52].
- Realization refers to elaborating the services into detailed implementation architecture.

2) **Detailed design support:** This criterion refers to whether the method supports detailed SOA service design models. Detailed design implies elaborating high-level service specifications models into a more detailed description of the service behaviour, interfaces, and data structures [29]. This activity is related to the specification design stage [8]. When detailed design is supported, the method specifies service interface

TABLE 7. Publication date.

Publication year	Articles
2007	1
2008	3
2009	8
2010	1
2011	6
2012	2
2013	3
2014	3
2015	2
2016	1
2018	1
2019	2
2020	3
2021	2
2022	1
2023	1

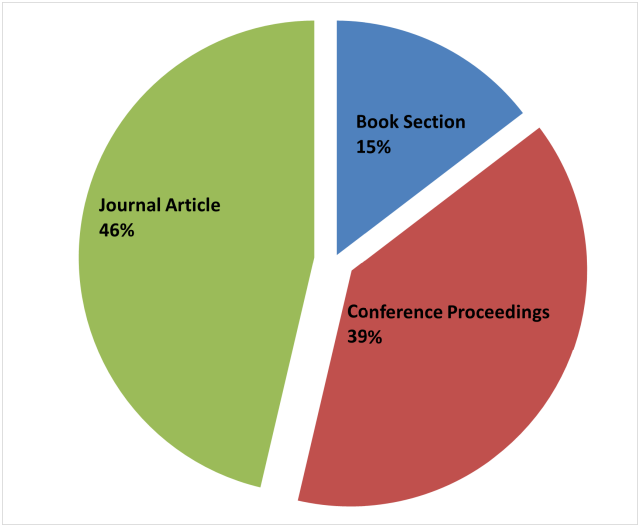

FIGURE 3. Papers type distribution.

details, including methods, parameters, and message formats. Deriving detailed service specification models helps align service models with supported PMs.

3) **SoaML support:** This criterion refers to whether the method supports the Service-oriented architecture Modeling Language (SoaML) [53]. SoaML is an OMG specification that provides a meta-model and a UML profile with service modeling capabilities to support designing SOA-based solutions. SoaML extends UML in six areas to define essential modeling services, including capabilities, protocols or rules, and information exchanged between consumers and providers (*Participants*) [53]:
   - Participants are physical or moral entities that provide or consume services and can be software components. They communicate using ports and can *provide or consume* any number of services. When using a service, the Participant is considered a *'Service Consumer'* and uses a port with a *'Request'* stereotype. When offering a service, the *Participant* is regarded as a *'Service Provider'* and will have a port with a *'Service'* stereotype.
   - Service Interface defines how a participant interacts to provide or consume services. It describes the interface and the Participant's responsibilities using the Service or Request port.
   - Service Contract specifies the agreement between service providers and consumers. It shares information between the Participants regarding products, values and resources without describing how and why they will fulfill their obligations. Thus, it enforces the SOA loose coupling principle.
   - Service Data describes the type and the content of messages.
   - Services Architectures describe how service consumers and providers collaborate. It expresses the dependencies between Service Contracts and the roles of the Participants.
   - Capabilities are used to model the ability to act and produce an outcome. In other words, capabilities identify and specify *the functions or resources* a service might offer.
4) **The use of patterns:** This criterion asserts whether the method uses patterns to identify and/or specify the services. A pattern is a reusable formalized solution that solves a frequent problem occurring within a specific context [5]. Patterns can be categorized into various types based on their use and context, such as business patterns [39], [54], [55], workflow patterns [56], [57], or design patterns [58].
5) **Service granularity:** This criterion assesses whether the method supports the service granularity principle when designing services. The term ''service granularity'' refers to the extent of the functionality that a service provides [59]. Service granularity is one of the major concerns during the service design [48]. It contributes significantly to the quality of the service [60], [61], [62]. Hence, designing services with the right granularity avoids design anti-patterns such as ''Tiny Service'' and ''God Object Service'' [5]:
   - ''Tiny Services'' are fine-grained services that provide a small amount of business-process utility, such as basic data access [5]. Having many fine-grained services impacts the system's performance [63].
   - ''God Object Services'' are coarse-grained services from a large composition of smaller-grained services [5]. The service granularity attribute can affect its capabilities, performance, and reuse [9].
6) **Level of automation:** This criterion assesses whether the method proposes techniques that can be automated. Lack of automation means that activities must be manually conducted, increasing the dependence on user involvement. This criterion can have three values: manual, semi-automated, and automated.

TABLE 8. List of papers, journals, and books.

Book series	**6**
Advanced Design Approaches to Emerging Software Systems: Principles, Methodologies and Tools	1
Communications in Computer and Information Science	1
Lecture Notes in Business Information Processing	1
Service Engineering: European Research Results	1
Lecture Notes in Computer Science	2
Conference Proceedings	**16**
2009 Second International Conference on the Applications of Digital Information and Web Technologies	1
IFIP Advances in Information and Communication Technology	1
Neonatal, Paediatric and Child Health Nursing	1
Proceedings of the ACM Symposium on Applied Computing	1
Proceedings of the Annual Hawaii International Conference on System Sciences	2
2008 International Workshop on Service-Oriented Computing: Consequences for Engineering Requirements	1
51st Hawaii International Conference on System Sciences	1
2008 IEEE Congress on Services Part II	1
2009 - 13th Pacific Asia Conference on Information Systems: IT Services in a Global Environment	1
16th Americas Conference on Information Systems 2010	1
15th European Conference on Information Systems	1
2nd World Conference on Complex Systems 2014	1
2009 IEEE International Conference on Networking, Sensing and Control	1
6th IEEE International Symposium on Service-Oriented System Engineering 2011	1
18th CSI International Symposium on Computer Science and Software Engineering 2017	1
Journal Article	**19**
Computer Science and Information Systems	1
IEEE Access	1
IEEE Transactions on Services Computing	2
IFIP Advances in Information and Communication Technology	1
Information Systems and e-Business Management	1
International Journal of Cooperative Information Systems	1
International Journal of Organisational Design and Engineering	1
International Journal of Software Engineering and Knowledge Engineering	1
iSys - Brazilian Journal of Information Systems	1
IT Convergence Practice	1
Journal of Applied Sciences Research	1
Journal of Systems and Software	1
Journal of Theoretical and Applied Information Technology	1
Journal of Web Engineering	1
Software and Systems Modeling	1
Software: Practice and Experience	1
Technical Report No. 1, TR-ASER-2012	1
International Journal of Web Services Research	1
Total of publications	**41**

7) **Tool support:** This criterion evaluates whether the method offers tools for the design process. Methods outline the necessary steps and provide instructions for users to carry out service design activities. Supporting tools implement the logic and techniques of the method, including algorithms and transformation rules. These tools help automate the design process and guide users, reducing the likelihood of human error. Therefore, the availability of tools dramatically improves the method's usability (i.e., enhancing its effectiveness, efficiency, and user satisfaction).

C. ANALYZING AND SYNTHESIZING THE RESULTS

To answer the corresponding research question (see Table 2), we analyzed and compared the methods using each of the seven criteria defined and justified in the comparison framework.

1) SOA SERVICE DESIGN LIFE CYCLE COVERAGE

The analysis of this criterion answers the research question RQ1 (*What are the SOA service design phases covered by existing methods?*).

The collected data reveals that all selected methods cover at least the service identification phase (see Table 9). Figure 4 portrays the design life cycle coverage distribution among selected methods. We found that 23 methods (i.e., 59% of the total) are limited to identifying services. The main objective of service identification is to extract and determine which services are appropriate to build the SOA-based solution from the business model. The output of this phase consists of a list of candidate services.

Conversely, ten methods (i.e., 28% of the total) cover service identification and specification phases. Those methods are found in the works of [1], [30], [60], [62], [68], [73], [76], [79], [89], and [93]. These methods include activities that identify candidate services from business models and then generate service specification models. Yet,

TABLE 9. Methods comparison criteria.

	Design life cycle coverage			Detailed design	Granularity	SoaML support	Pattern-based	Tool support	Automation
	Identification	Specification	Realization						
[64]	X			X					M
[65]	X								M
[66]	X								SA
[67]	X								M
[33]	X						X		M
[68]	X				X		X		M
[69]	X				X			X	SA
[70]	X				X		X		M
[1]	X	X		X	X	X	X		SA
[62]	X	X			X				SA
[71]	X				X				SA
[2]	X	X	X						M
[61]	X	X	X	X	X	X	X	X	SA
[72]	X				X			X	SA
[73]	X	X		X		X			SA
[74]	X								M
[75]	X	X	X		X				M
[32]	X				X				SA
[59]	X				X			X	SA
[76]	X	X			X				SA
[77]	X				X				SA
[78]	X								M
[79]	X	X		X					M
[60]	X	X		X	X		X		SA
[80]	X				X				SA
[28]	X	X	X	X	X			X	SA
[81]	X				X				SA
[34]	X	X	X				X	X	M
[82]	X	X	X						M
[83]	X	X	X						M
[84]	X								M
[85]	X								M
[86]	X				X		X		M
[30]	X	X				X	X		SA
[87]	X								M
[88]	X	X	X		X	X			SA
[89]	X	X					X		SA
[90]	X				X				SA
[91]	X							X	A
[92]	X	X	X			X		X	A
[93]	X	X			X			X	SA

X = method includes the criterion. Empty slot = criterion is not met. A = Automated. SA = Semi-automated. M = manual.

the generated service models (output models) differ in the modeling notation/language and the level of the architectural specification details.

Lastly, we identified nine methods (i.e., 13% of the total) covering three service design phases (i.e., identify, specify, and realize/elaborate service models). Those methods are found in the works of [2], [28], [34], [61], [75], [82], [83], [88], and [92]. These methods comprehend service realization activities that transform service specification into implementation models. They utilize various SOA-related service technologies such as Web Services, microservices, REST, or Business Process Execution Language (BPEL). Furthermore, authors in [28], [61], and [92] claim that their methods generate executable code by processing a model-to-text transformation from generated service models.

To answer the RQ1 (*What are the SOA service design phases covered by existing methods?*), our findings show that selected methods are in three main categories: i) Identification only methods (59%), ii) Identification and specification methods (23%), and iii) Identification, specification, and realization methods (13%). However, all of the methods cover at least the identification phase. Various approaches and techniques are used to identify candidate services during this phase.

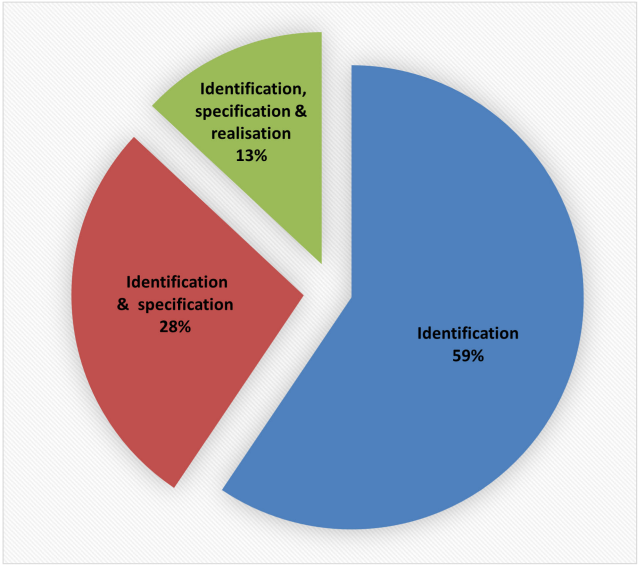

FIGURE 4. Distribution of the design life cycle coverage.

2) DETAILED DESIGN SUPPORT

The analysis of this criterion determines if the method supports detailed service specification design and answers the research question RQ2 (*What are the methods that support detailed service specification?*).

Extracted data shows that even if a method covers the specification stage, only six of the selected methods support detailed service specifications. Thus, we identified those methods in the works of [1], [28], [60], [61], [75], [88], and [93]. Our analysis describes each method's various approaches to generate detailed specification models.

In [75], the detailed design activities address the specification of "Business Objects" (BO)[4] and of "Service Operations" (SO). First, BOs are extracted from the message exchange within the PM. BOs are broken down into information and structure components. Identified components are consolidated in a single table. Based on that table, the structure components are mapped to business objects, while the information components serve as attributes of the business objects. Further analysis of the PM generates a mapping table between actors, identified business transactions,[5] and required activities. This mapping helps to consolidate activities into services and specify their operations.

The method in [88] proposes analyzing exchanged messages within the business process to specify the service interfaces. Interface input and output parameters are identified based on activities' operations and the variables from the data object in the PM. If an activity accesses or removes a variable, it is set as an input parameter. If an activity initializes or updates a variable, it is set as an output parameter.

[4]Business objects are used to illustrate how data flows through a process and how it is transformed or manipulated as the process progresses.

[5]A business transaction is a logical unit composed from a sequence of activities that affect a business relationship between two or more actors.

The method in [93] drives two types of services: business and web services. Service-related functionality, behaviour, and policy are specified using two ontological models defining the business and web service models. The business service model is defined by extending the collaboration-oriented ontology OeBTO to capture service-related functionality and service quality requirements. The Ontology Web Language (OWL) [94] enables the validation of the model consistency and automatic transformation to the web service level.

In [61], the method uses workflow patterns from the business process models to derive the service interfaces, contracts, and participants' detailed design. In [60], the method determines the structure and behaviour of a service using action patterns from business process models. Action patterns describe recurring behaviours in process models.

On the other hand, [28] uses syntactic and semantic analysis to extract relevant information from the process models to derive the service details. Service operations are specified using REST Verbs (GET, POST, PUT, DELETE) when the data is unavailable. However, the outcome of this approach depends heavily on the accuracy of synonyms. In [1], the authors use structural and behavioural business patterns from [39] and [54] to specify identified services accurately. They propose specifying the SOA service interfaces (one-way, bi-directional), participants, architecture, and choreography using the SoaML language.

> To answer RQ2 (*What methods support detailed service specification models?*), our findings show that six out of 41 methods support detailed service specification. The methods employ different techniques to extract information to infer the attributes, structure, and operations, allowing the specification of identified services using, for example, interfaces (provided, required), dependencies, service architecture, and participants.

3) SOAML SUPPORT

This criterion verifies whether the method uses the SoaML language for service modeling. Analyzing this criterion allows us to answer the research question (RQ3) (*What are the methods that support SoaML in describing the services?*).

To compare the methods from this criterion's perspective, we verified the modeling language and the techniques used to generate specification models of the services. Thus, we identified six (6) out of 41 methods that support and use SoaML in our selection. However, each method uses SoaML modeling capabilities differently.

The method proposed in [73] derives service specification from business models as a set of SoaML model artifacts. Hence, the authors combine different SoaML modeling approaches. The first step uses a Service Contract approach

for business-level architecture modeling. Participants' models are derived from the business process elements. A Service Contract model defines the ''Roles'' that each Participant plays in the service (i.e., Provider and Consumer) and the Interfaces they implement to play that Role. A Behavior model describes the Interactions between the Roles in a Service Contract. A Service Architecture model comprises all identified Service Contracts and Participants. The Services Architectures are modeled as UML Collaborations. The second step uses a Service Interface approach for the system-level architecture modeling. Specified Service Contract models are refined to Service Interface models. The Service Contract model defines the Interface and responsibilities of the Provider and the Consumer, including commands and information to initiate the actions. To model the Service Interfaces, the method uses two approaches: i) a Simple Interface-based approach that specifies a uni-directional service focusing on a one-way interaction provided by a Participant and ii) a Service Interface-based approach that specifies a bi-directional service, where ''Callbacks'' exist at the Provider's side.

The method in [30] uses heuristics to generate SoaML reference architecture for software services using a mapping of KAOS goal model concepts to SoaML concepts. For instance, candidate services are transformed into SoaML Service Contracts, and KAOS Agents are transformed into SoaML Participants. The method proposed by [61] generates SoaML models, including SoaML Services Architecture, Participants (Provider and Consumer), Service Interfaces, and Service Contracts. The modeling process uses an ontology that defines mapping rules between corresponding Meta-model elements of the BPMN and SoaML specification. The approach from [92] implements transformation rules to derive SoaML service models from ERP exchange and conversion processes. The method uses a set of mapping rules between the BPMN Meta-model and SoaML Meta-model constructs.

SoaML is also used to specify services in [88]. The service specification phase provides detailed service models using Messages, Service Interfaces, Service Contracts, and Services Architecture SoaML models. Finally, the proposed method in [1] uses SoaML to generate Participants, Service Behaviour, and Interface models. The method uses either a Simple Interface approach for Uni-directional services or a Service Interface approach for Bi-directional services (i.e., services with callbacks from the provider to the consumer).

To answer RQ3 (*What are the methods that support SoaML in describing the services?*), data reveals that 19 methods (41%) generate architectural service specification models. Generated service models are expressed using various modeling languages (UML, SoaML, XML, REST, or BEPL). Only six methods support the SoaML language (i.e., 31% out of 19 methods that cover the service specification phase).

4) THE USE OF PATTERNS

We use this criterion to verify whether the method uses patterns to answer the research question RQ4 (*What are the methods that use formalized solutions, such as patterns, to identify and/or specify the services?*).

To answer this research question, we examined the nature of approaches adopted by existing methods. The data collected from this SLR reveals that selected methods use three various patterns: business patterns, workflow patterns, and design patterns. Business patterns refer to a set of solutions to solve common business concerns [54]. Workflow patterns are specific business patterns that refer to common structures and interactions found in business processes [56], [57]. Design patterns are formalized solutions to a frequent design problem ([58], [95]).

Within this SLR selection, we identified seven (7) methods that use patterns during the service design process. These methods use various techniques. Authors in [89] generate service models by translating the Workflow/Control-flow patterns[6] found in the PMs, into a service orchestration model.

In [30], [34], [61], [68], [70], and [86], services are identified or specified using workflow patterns.

On the other hand, we identified two methods that use business patterns. In [33], the method detects recurring business patterns within business value models to i) derive business process models and then ii) extract the services. On the other hand, authors in [1] use business ontology patterns (see [38], [39], [54]) detected in the BPMN choreography model to specify the identified services. Each choreography task is mapped to at least one business pattern. To map the tasks and the patterns, the authors use the Open EDI reference model [96]. Finally, the method proposed by [60], uses action patterns to specify the structure and behaviour of identified services. Action patterns are specific design patterns [95].

It is worth noting that other methods use a formal-based approach[7] to identify or specify the services. However, we reported that authors in [34] use Perti-net formal notation to verify whether elements of the BPMN model satisfy semantic preconditions before being translated into BPEL constructs.

To answer RQ4 (*What are the methods that use formalized solutions such as patterns to identify and/or specify the services?*), our findings show that ten (10) methods (24%) out of 41, use patterns, namely business patterns (4%), work-flow patterns (17%) and design patterns (2%).

[6]Control-flow is a perspective of the workflow to describe the activities and their order of execution using different constructors such as sequence, parallelism and join synchronization [56].

[7]Formal-based methods use rigorous techniques based on mathematical foundations (i.e. given a precise mathematical meaning) [97].

5) SERVICE GRANULARITY

The analysis of this criterion verifies whether the method considers designing services with the right granularity to answer the research question RQ5 (*What are the methods that consider the service granularity principle?*).

Analysis of data collected from this SLR reveals that 21 methods among the 41 selected for this SLR support the service granularity principle. Each method combines different approaches and techniques to optimize the quality attribute of service design.

In [64], the authors identify services by analyzing the business processes and activities. They describe elementary activities using a minimized use case template that determines the required functions. They consolidate the service functions by clustering ''logical function blocks'' based on i) the provided functionality and ii) the data structure on which they operate. However, the clustering process relies on expert knowledge. Finally, they generate a service description containing the consolidated functions' names, purposes, and descriptions.

The method presented in [81] identifies a fine-grained elementary service for each business entity. A graph partitioning algorithm uses the process activities to define the right service granularity level. Elementary services are clustered into composed services to support related business activities. Authors in [76] use a clustering technique based on an affinity analysis of a matrix composed of elementary business processes and business entities. They compose services with the right granularity by grouping elementary business processes that operate on the same business entities. In [90], the authors apply the entity affinity analysis clustering technique from [76] to group functions into services. The authors in [69] present an iterative algorithm to refine identified services according to metrics of the service quality attributes. The use of metrics allows for the definition of a suitable granularity level. The method proposed in [77] uses a genetic algorithm to ensure that identified services are optimized regarding their design quality attributes. Design metrics are defined based on the business goals for each quality attribute. To optimize service granularity level, the algorithm identifies Pareto solutions. In the context of multi-objective optimization, Pareto solutions are better solutions concerning at least one attribute. A set of Pareto solutions represents a set of services. Then, fuzzy logic is utilized to choose the set of services with a suitable abstraction level according to business goals. Fuzzy logic allows users to model imprecise or vague data [98]. Authors in [80] use a Hybrid Particle Swarm Optimization (HPSO) algorithm to optimize service granularity. The first step decomposes the business process into elementary business processes encompassing activities related to the same business entities. Then, they build a value-based matrix representing the impact (i.e., C.R.U.D) of the elementary business processes on business entities. The HPSO algorithm computes the semantic dependencies and affinities metrics to optimize the clustering phase of matrix elements into candidate services with the right granularity level. In [88], the authors identify each pair of process activities, relations regarding goals, business data, and process models. A genetic algorithm aggregates these relations to show the total dependency between the process activities. Each pair of activities is clustered into a service based on their total dependency. Authors in [62] use a multi-objective evolutionary algorithm to resolve the issue of conflicting metrics for service cohesion, coupling, and granularity attributes.

To define services with the right granularity, the method in [93] decomposes aggregated services into more granular services based on tasks to operate the consumption and acquisition of economic resources[8] through the collaboration process.

In [71], the authors present an approach for identifying microservices by analyzing business process elements' dependencies. Then, they apply clustering techniques to gather activities into fine-grained microservices. Authors in [68] and [86], use a set of heuristics to build a candidate services dependency graph. This graph is based on the relationships among the elements of the PMs (e.g., activities and flows) from which the candidate services are identified. Then, a granularity map for candidate services is built. Coarser-grained services are at the top level of the map, while finer-grained services are at the bottom. In [75], to obtain a suitable service granularity, authors suggest that a service must at least comprise all operations performed by the same actor and support all the operations to complete business transactions.[9] In [61], the method defines the service granularity from a single or grouped task in the business process models.

Authors in [60] identify activities and business objects from the process model. They use semantic analysis algorithms to derive both atomic and composite services. The similarity between the two activities is based on the semantic intersection between their actions/operations and the business objects. Semantically intersecting activities are aggregated into a single fine-grained atomic service candidate. Activity groups related to the same business object group define a composite service candidate. Business object groups are semantically identical business objects.

In [28], the method defines the RESTful service scope according to each task within the BPMN choreography business process models. The approach in [1] derives a service for each choreography task in the choreography business process model. The service granularity is defined based on the corresponding choreography task message (e.g., One-Way, Request-Response). Defining service models with the right granularity avoids the occurrence of service antipatterns[10] [5].

[8] Based on the REA ontology, economic resources are defined as objects under the control of an enterprise that are scarce and have utility [38].

[9] A business transaction is a logical unit composed from a sequence of activities that affect a business relationship between two or more actors.

[10] A service antipattern is a commonly used solution to solve a design problem that may increase the design complexity.

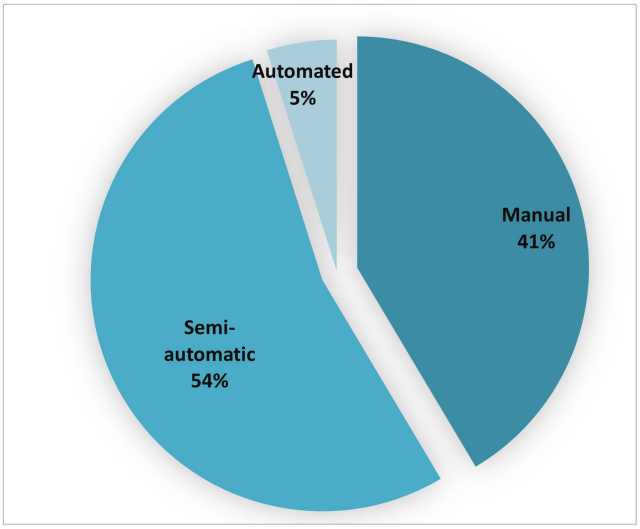

FIGURE 5. Automation usage among selected methods.

Authors in [72] identify and classify candidate services into different categories based on their granularity layers. Services in the upper layers are coarser-grained, while services in the lower layers are finer-grained (e.g., process services versus data or utility services). Birkmeier et al. [70] apply heuristics to produce a hierarchy of service candidates of different granularity levels. Reference [59] uses a service typology to guide service identification with the right granularity. The model defines four types of services: Utility services, Entity services, Task services, and Process services. Utility services provide shared functions unrelated to business logic (e.g., authentication, encryption, and logging). The functional scope of Entity services is used to manipulate one business entity. Task services contain specific business logic. Finally, Process services introduce a level of abstraction to manage interaction details required to ensure that service operations are executed in a specific sequence.

To answer the research question RQ5 (*What are the methods that consider service granularity principal?*), our findings show that 21 methods from this SLR selection use different approaches and techniques to derive services with the right granularity level.

6) AUTOMATION LEVEL

Analyzing this criterion allows us to classify the methods according to their level of automation to answer the research question RQ6 (*What are the levels of automation among the available methods?*).

Figure 5 shows a spectrum of automation levels among selected methods. We distinguish three types of methods: i) manual, ii) semi-automatic, and iii) automatic.

Several methods are limited to manual processing guided by specific instructions. Manual methods rely heavily on the designer's knowledge and skills. Those methods are found in the works of ([2], [33], [64], [65], [67], [68], [70], [74], [75], [78], [79], [82], [83], [84], [85], [86], [87]).

The semi-automated methods use automation techniques; However, they still require human judgment or intervention to guide and launch the process. Semi-automated methods are proposed in [1], [28], [32], [59], [60], [61], [62], [66], [69], [71], [72], [73], [76], [77], [80], and [93].

Authors in [91] and [92] claim that their methods are fully automated.

Further analysis of the methods reveals the variety of automation techniques. Automation techniques use an algorithm to partition, cluster, or aggregate process tasks and activities into services. The methods that use such algorithms to identify services are found in [32], [62], [69], [71], [76], [77], [80], [81], [90], and [99]. Authors in [30], use an algorithm that applies a set of heuristics to identify and specify services from a GM. Other methods use algorithms to automate the Model-to-Model transformations by applying predefined mapping rules between elements of the business models and their corresponding elements of the service models. These techniques are used in the method proposed by [59], [61], [73], [88], [89], [92], and [93]. The methods proposed by [28] and [34], use automation algorithms to perform semantic and syntactic analysis to detect specific patterns that support the service specification and design tasks. Likewise, authors in [1] use a semi-automated service design method based on minimal domain expert involvement to annotate the BPMN choreography business process models.

To answer the research question RQ6 (*What are the levels of automation among the available methods?*), we identified 41% manual methods. The remaining methods are either semi-automated (54%) or automated (5%).

7) TOOL SUPPORT

The analysis of this criterion verifies whether the method has a supporting tool to answer the research question RQ7 (*What are the methods that offer supporting tools?*).

The data on selected methods indicate that only a few methods provide tools for identifying and specifying services from business models.

In [72], authors presented a tool for service identification. The method in [34] implemented an online tool to support the method's transformations from BPMN to BPEL. Authors in [59] developed a tool to support identifying business entities, tasks, and process services.

The authors in [93] used OWL design-supporting tools, namely XQuery and XML transformations and XSLT technologies, to enforce the automation of the model transformation.

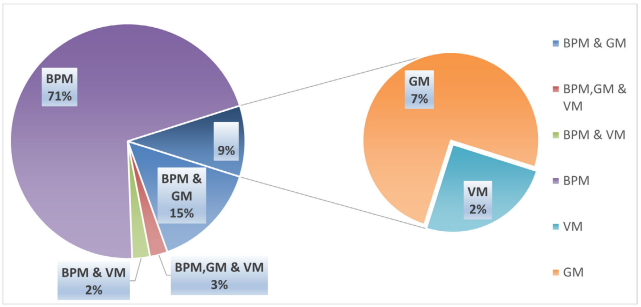

FIGURE 6. Input business model types.

In [69], authors developed a Java-based application to support the service identification activities of the method. Authors in [61] developed an application to handle service modeling and model-to-model transformations. This tool is based on the Eclipse Modeling Framework (EMF). It includes a QVT transformation engine, the Eclipse SoaML plug-in for service modeling, and the Eclipse SoaML2Code plug-in to generate the code. Authors in [92] use a transformation tool based on Eclipse ATL and Acceleo modules to perform Model-to-Model (M2M) and Model-to-Text (M2T) transformations. In [28], authors developed the REST Annotator tool to extract and analyze task labels. REST Annotator allows the verification of similarities with REST verb synonyms and generates a set of choreography diagrams enriched with REST annotation.

Reference [91] developed an application to support clustering algorithms for identifying candidate services. In addition, the tool offers a graphical interface that shows identified clusters as service candidates.

To answer the research question RQ7 (*What are the methods that offer supporting tools?*), we identified nine methods (i.e., 22%) of selected studies that provide tool support.

D. DISCUSSION

In the previous subsection, we used a set of quality criteria to analyze and compare existing service design methods. Our findings were used to answer each research question from this SLR (see Table 2). As far as we observed, each method was designed around various purposes, approaches, and techniques.

Our analysis of input models used by the methods indicates that PMs are the dominant model type. Figure 6 shows the distribution of input model types used by selected methods. We noticed that 37 methods (91%) use PMs as inputs. Among these methods, six combine PM with GM as input models. One method uses a PM and a VM. One method uses all three model types. The remaining four methods use either a VM (one) or a GM (three) as input models. According to [60], their availability could explain the prevalent usage of PMs as an input model. However, scholars have outlined the contribution of GMs and VMs to reducing the gap between service and business models [30], [33], [59], [93]. They argue that PM, GM, and VM models provide different yet complementary views of business processes. The GM articulates the motivation for the business process, while the VM reveals its value chain. Using multiple business models for service identification provides a comprehensive view that aligns services with an organization's business processes. This approach captures various business perspectives from the business models to enhance SOA service flexibility and adaptability during shifting priorities. It also allows service designers to find opportunities for modularization and reuse of services, improving SOA efficiency. Additionally, combining different business models balances fine-grained and coarse-grained services, resulting in functional and maintainable offerings that are technically feasible and relevant to business objectives. However, despite all the benefits mentioned above, using multiple models as input requires more coordination and adapted techniques during the service design steps.

All selected methods from this SLR cover at least the service identification stage. We noted that 22 methods from this SLR focus on solving specific service identification problems without further details about how the service models will be specified. On the other hand, other methods generate service specification models described with different levels of detail, ranging from a simple and informal list of candidate services to more detailed service descriptions. In summary, a service design method that includes identification and specification offers a structured approach for developing service models aligned with business models. This method ensures technical accuracy and comprehensive documentation, facilitating the transition from service identification to implementation. By maintaining consistency between service roles and their definitions, the method's design stage coverage of identification and specification enhances service quality and precision in accordance with SOA principles. This leads to more straightforward implementation, maintenance, and scaling of services while reducing redundancy and alignment challenges. Ultimately, these practices result in adaptable and scalable service-based solutions.

One of the main objectives of the proposed methods is to simplify the service design and reduce the time and effort needed to derive services from business models. Analyzing the techniques reveals that selected methods propose various approaches to identifying services from business models. These methods employ a domain decomposition strategy to break the business model into smaller logical units, such as Tasks/Activities, Messages, Workflows, Values, and Resources [8]. When using this domain decomposition process, it is essential to find an optimal way to map the identified business logic units from the business models into candidate services. In the context of SOA, this mapping should consider design quality principles, including cohesion, coupling, and granularity. Consequently, the methods also explore

various clustering techniques. Clustering involves defining the scope of each service to optimize its quality, including the appropriate level of granularity.

Service granularity is one of the service design challenges identified by authors in [6]. This study's data found 22 methods (49%) that support the service granularity principle. We summarize our findings about the techniques used to support the service granularity principle as follows:

1) Business model decomposition via an iterative process to attain an optimal partitioning of business capabilities into services [100]. However, candidate services extracted from domain decomposition without further refinement are likely to generate SAO design antipatterns such as "God Object Services" or "Tiny Services" (see [5]).
2) Service classification is performed after the domain decomposition activities. Thus, designers classify services into different typological schemes or a hierarchical taxonomy [101]. Service hierarchies are organized from fine-grained to coarse-grained services. Atomic services are composed to form new coarse-grained services. Service typologies are based on business process elements such as activity and task services, entity and data services, and utility services [9]. Their granularity level is defined according to their related business process element (e.g., task, data).
3) Clustering of services is done by using optimization algorithms, dependency graphs, and CRUD matrices. These techniques aim to optimize the dependency ratios among identified services by utilizing structural and semantics relationships between business model elements such as tasks, entities, and goals. According to [62], service granularity optimization algorithms fall into two categories: deterministic and nondeterministic. Deterministic algorithms present a specific service design (e.g., [69], [76]). Nondeterministic algorithms use heuristics or meta-heuristic optimization algorithms (e.g., genetic algorithm or particle swarm optimization used in [77], [80], and [88]). Using optimization algorithms requires the definition of metrics to guide the design process in shaping high-quality services in accordance with SOA design principles. Thus, the performance of the used algorithms depends on well-defined inputs and the quality of the business process analysis.
4) Semantic and syntactic analysis of the business models' textual and structural data is conducted to detect similarities and patterns. Tasks and process activities are clustered into services based on their affinity. However, this technique depends on the accuracy of the synonyms.
5) Building a choreography business model. The granularity level is set to choreography tasks.
6) Using business patterns or workflow patterns to determine the service scope of functionalities.

Optimizing service granularity of identified SOA services improves performance, reusability, scalability, maintainability, and alignment with business goals [60]. This optimization allows services to function in a balanced, efficient manner, enhancing the overall agility and responsiveness of the service-based solution while reducing operational costs and complexity. However, each one of the techniques used by the available methods has its strengths and weaknesses [62]. These techniques can be combined, optimized, and refined to achieve a better design outcome.

Supporting detailed specifications of service models is an important quality criterion for design methods. Service specification models define a service's functionality, interface, and requirements at different levels. Service specifications provide high-level requirements and an overview of the service, including its inputs, outputs, and primary functions, without detailed implementation information. In contrast, detailed specifications comprehensively define functionality and technical aspects, including exact data formats, protocols, error handling, security requirements, and performance constraints. These are designed to guide developers during the implementation and integration process, ensuring consistency and completeness.

Existing methods use various techniques that permit the derivation of detailed models which specify the services' structures and behaviour. Our findings show that only 15% of the selected methods provide techniques to support such design detail. Yet, we observed that when using syntactic and semantic analysis (like in [28] and [60]), the quality of the generated service model depends on the accuracy of synonyms. Moreover, when using a model-to-model (M2M) transformation or specifying service details based on the default CRUD operations (such as in [32], [59], and [79]), the generated service models lack business semantics. Conversely, using an approach that adopts the design best practices, such as patterns[11] could enhance the design quality of the identified service models. Patterns are designed with flexibility, enabling systems to adapt more easily as business needs evolve. They provide effective solutions in a structured and reusable manner, enhancing both the reusability and efficiency of the services.

Another criterion that impacts the quality of the method is the adoption of SoaML to support the activities of service modeling and design. SoaML provides the means to generate a comprehensive service architecture model that provides details about the service attributes, interfaces, messages, and detailed operations. Methods that generate service models using SoaML leverage its support of both the technical and the business perspective by describing the capabilities of organizations, communities, and systems [53]. We found that 32% of the existing methods have adopted SoaML since its standardization by the OMG in 2011. However, we noticed that the methods do not utilize the full potential of SoaML to generate SOA architectural models.

[11] Patterns are good solutions to recurring problems [5].

Automation of design tasks is a challenge for existing methods. Automation uses different techniques, including algorithms, transformation tools, and heuristics. Algorithms are used to automate service quality attribute optimization, semantic and syntactic analysis, pattern detection, and rule-based transformations. Even if recent methods are invested in implementing automated and simplified design techniques, manual tasks and complex approaches are still used. Manual tasks involve the designer's know-how and costly resources, leaving room for human errors to affect the quality of the extracted services. Thus, adopting automation techniques improves the quality of the methods. However, when kept to a minimum, balanced user involvement is needed to bring the business domain inputs to align generated service models with supported business models.

Furthermore, providing tool support helps users perform design activities in a structured and automated manner while avoiding human errors. A tool facilitates the implementation of the method and its adoption by the organization. Summarizing the above discussion, as observed in this SLR, existing methods still cover the design stages only partially and do not adopt design best practices, automation techniques, or tool support. Therefore, improvements are required to enhance their usability and design quality.

Although some studies have focused on specific issues like granularity optimization, service identification, and automation, a closer look at the data from this SLR highlights evolving trends among recent proposed methods. New methods tend to increasingly adopt more of the quality criteria identified in this study. However, future methods must be fully implemented, and quality criteria must be combined to improve service design. For instance, effectively covering the service design cycle requires detailed specification support to realize its full benefits. This also involves optimizing service granularity and supporting the SoaML language to enhance the quality of service models.

Furthermore, utilizing best practices such as patterns can improve the alignment of derived service models with business models. Patterns offer several advantages, including reusability and flexibility [5]. The use of SoaML and patterns simplifies method design tasks and promotes agility. Additionally, automation and tool support are essential for enhancing usability. These tools offer practitioners–such as solutions architects, business architects, and application architects–an easy-to-use interface. method for deriving SOA architectural models from business models.

VIII. THREATS TO VALIDITY

Threats to validity in a systematic literature review (SLR) can compromise the credibility of the findings. We identified multiple validity threats to this SLR, focusing on the accuracy of the findings within the context of the reviewed studies. The identified threats are related to selection bias, inadequate search strategy, publication bias, data extraction, interpretation bias and temporal bias. Addressing these threats is crucial to ensuring the rigor, reliability, and generalizability of the findings.

A. SELECTION BIAS, INADEQUATE SEARCH STRATEGY, AND PUBLICATION BIAS

Threats related to selection bias occur when studies are selectively included or excluded based on subjective preferences or unintentional biases. This can lead to non-representative findings, the omission of relevant studies, or the inclusion of less relevant ones. To mitigate selection bias, we cast a wide net, using a comprehensive range of databases related to studies in information systems management and computer science. Furthermore, we predefined explicit inclusion and exclusion criteria in a well-defined protocol and applied them consistently across the studies. We also screened the search strategy by peer reviewÂ to ensure it covered all relevant sources.

Threats from an inadequate search strategy arise from an incomplete search strategy that might miss relevant studies, leading to a non-representative selection. Publication bias threats occur when studies with significant or favourable results are more likely to be published, which can skew results. Thus, due to the variety of search strategies, the use of heterogeneous keywords, and the limitations of using a lengthy search string, we faced some challenges in finding possible related studies. To overcome this, we defined a comprehensive and transparent search strategy and used the reference management guidelines of each database. Our search strategy was enriched by using various synonyms and acronyms of the keywords to build customized search strings and running multiple query tryouts using multiple databases to find the most significant number of related studies. In addition, we used cross-referencing to perform forward and backward snowballing searches and extend our selection to any additional studies we may have missed in the database search results.

B. DATA EXTRACTION BIAS

Common threats from data extraction bias are variations in how data is extracted, which can lead to inaccuracies or selective emphasis on specific results. To minimize this risk, we fostered a collaborative environment where multiple reviewers performed data extraction independently. A standardized data extraction process was used to ensure consistency in capturing relevant information from each study. Any discrepancies in data extraction were discussed and resolved through consensus among the reviewers. This approach reduces bias and errors, ensuring a comprehensive and balanced review.

C. TEMPORAL BIAS

Other threats to this SLR finding are contextual and temporal limitations. The findings from this SLR may be influenced by temporal bias since the time frame of the included studies covers nearly two decades. To mitigate these threats, we conducted a temporal analysis to examine trends and

identify whether there are any significant changes in the field. This approach helps ensure that the review reflects historical and current developments in the field of SOA service design. It is worth noting that related studies published after our selection process are missing.

D. INTERPRETATION BIAS

Interpreting the findings can be subjective, leading to inconsistencies and potential bias in the results. In this SLR, we carefully interpreted our findings within the limitations of the selected studies to mitigate the threats of interpretation bias. This study is limited to methods for service identification derived from business models. Therefore, our findings cannot be generalized to all SOA service identification method types.

Incorrectly inferring causality from available data is an identified threat that can compromise the accuracy of conclusions drawn from this SLR. We admit that some variations and inconsistent definitions of SOA concepts used by scholars can threaten the validity of our synthesis. However, to mitigate this, we relied on well-established concepts from SOA principles and frameworks [20], [23], [47], [53]. These concepts provide a solid foundation for discussion and interpretation of our findings. Additionally, we focused on specific quality criteria to compare the methods and ensure the pertinence of our findings. However, we don't pretend that our criteria selection covers all aspects of an efficient SOA service design method.

In summary, addressing the above-mentioned threats helps enhance the validity of this systematic literature review, ensuring the findings are accurate, reliable, and unbiased.

IX. CONCLUSION AND FUTURE WORK

The emergence of SOA architectural style is a turning point for IS design and development [19]. SOA supports technical and business perspectives while defining quality design principles to promote the alignment between business models and service-based software solutions [23]. Deriving quality services from business models is a complex task that presents several challenges [6]. To overcome these challenges, several methods were proposed to identify and specify quality services from business models. In this work, we used a structured methodology to conduct an SLR about the methods that derive SOA services from business models. While this SLR surveys existing methods to derive SOA models from business models, it also provides architecture practitioners (e.g., solutions architects, business architects, and application architects) with tools to build effective SOA-based software solutions.

We selected forty-one articles from the literature and compared their characteristics according to selected quality criteria. We used the results from the SLR to answer the research questions. Our results show that existing methods present several limitations concerning the selected quality criteria. To summarize, our conclusions are fourfold:

- The existing methods offer different SOA design life cycle stage coverage levels. Hence, to enhance the quality of the design process output and promote the alignment between service models and their supported process model, a method must provide comprehensive process identification tools and support the detailed specification of services. Detailed specification models include details about the structure and behaviours of the services.
- A lack of automation and reliance on user involvement justify the need to design a method that requires minimal contribution from the designer's know-how to avoid human errors [15]. The novel approach must consider design best practices and easy-to-use techniques to identify and specify service models.
- Service granularity affects the quality of the architecture design of SOA services [60]. To improve service design quality, methods must define an optimal granularity level while avoiding anti-patterns and promoting reuse.
- The methods deriving services from business models promote the reuse of the business knowledge and semantics embedded within the models [29], [33]. This will better align designed services and their supported business processes [102]. Hence, linking business goals with generated service models will help the organization to evaluate the impact of derived services on achieving the business goals [30], [33].

This paper's findings will assist both the industry and academia in identifying current challenges and planning future research directions. It sets the scope for research projects to develop a novel method for deriving services from business models and helping organizations support their processes to achieve their business goals. Moreover, future research should consider creating new tools and methodologies that utilize and combine the set of quality criteria from this SLR to enhance the service design process and its outcomes. The expected method will ease the complexity of the current design process, enabling service-based solutions to meet the SAO principles for service design. Consequently, it will help business and industrial applications while bridging the gap between IS solutions and business requirements.

REFERENCES

[1] S. Alter, "A model-driven method to design SoaML services from BPMN models: Principles, proof-of-concept, and validation," in *Proc. 56th Hawaii Int. Conf. Syst. Sci.*, Maui, HI, USA, T. X. Bui, Ed., Jan. 2024, pp. 5799–5808.

[2] V. De Castro, E. Marcos, and R. Wieringa, "Towards a service-oriented MDA-based approach to the alignment of business processes with IT systems: From the business model to a web service composition model," *Int. J. Cooperat. Inf. Syst.*, vol. 18, no. 2, pp. 225–260, Jun. 2009.

[3] M. P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann, "Service-oriented computing: A research roadmap," *Int. J. Cooperat. Inf. Syst.*, vol. 17, no. 2, pp. 223–255, Jun. 2008.

[4] D. Ameller, X. Burgués, O. Collell, D. Costal, X. Franch, and M. P. Papazoglou, "Development of service-oriented architectures using model-driven development: A mapping study," *Inf. Softw. Technol.*, vol. 62, pp. 42–66, Jun. 2015.

[5] F. Palma, N. Moha, and Y.-G. Guéhéneuc, "UniDoSA: The unified specification and detection of service antipatterns," *IEEE Trans. Softw. Eng.*, vol. 45, no. 10, pp. 1024–1053, Oct. 2019.

[6] B. Bani-Ismail and Y. Baghdadi, "A literature review on service identification challenges in service oriented architecture," in *Knowledge Management in Organizations* (Communications in Computer and Information Science), vol. 877, L. Uden, B. Hadzima, and I.-H. Ting, Eds., Cham, Cham, Switzerland: Springer, 2018, pp. 203–214.

[7] F. Rowe, "What literature review is not: Diversity, boundaries and recommendations," *Eur. J. Inf. Syst.*, vol. 23, no. 3, pp. 241–255, May 2014.

[8] S. Cai, Y. Liu, and X. Wang, "A survey of service identification strategies," in *Proc. IEEE Asia–Pacific Services Comput. Conf.*, Dec. 2011, pp. 464–470.

[9] R. S. Huergo, P. F. Pires, F. C. Delicato, B. Costa, E. Cavalcante, and T. Batista, "A systematic survey of service identification methods," *Service Oriented Comput. Appl.*, vol. 8, no. 3, pp. 199–219, Sep. 2014.

[10] R. Boerner and M. Goeken, "Methods for service identification: A criteria-based literature review," in *Proc. 7th Int. Workshop Model., Simul., Verification Validation Enterprise Inf. Syst.*, Milan, Italy, D. Moldt, J. C. Augusto, and U. Ultes-Nitsche, Eds., 2009, pp. 76–84.

[11] Q. Gu and P. Lago, "Service identification methods: A systematic literature review," in *Towards a Service-Based Internet* (Lecture Notes in Computer Science), vol. 6481, E. Di Nitto and R. Yahyapour, Eds., Berlin, Germany: Springer, 2010, pp. 37–50.

[12] T. Vale, G. B. Figueiredo, E. S. de Almeida, and S. R. D. L. Meira, "A study on service identification methods for software product lines," in *Proc. 16th Int. Softw. Product Line Conf.*, New York, NY, USA, Sep. 2012, pp. 156–163.

[13] T. Kohlborn, A. Korthaus, T. Chan, and M. Rosemann, "Service analysis—A critical assessment of the state of the art," in *Proc. 17th Eur. Conf. Inf. Syst.*, Verona, Italy, S. Newell, E. A. Whitley, N. Pouloudi, J. Wareham, and L. Mathiassen, Eds., 2009, p. 1583.

[14] A. T. Zadeh, S. Sahran, and M. Mukhtar, "Automated service identification methods: A review," *Int. J. Adv. Sci., Eng. Inf. Technol.*, vol. 6, no. 6, pp. 1053–1059, Dec. 2016.

[15] T. Fausel and N. Hussein, "Approaches of service identification: Selective comparison of existing service identification methods," *Global Bus. Finance Rev.*, vol. 23, no. 4, pp. 46–74, Dec. 2018.

[16] F. Kohlmann and R. Alt, "Business-driven service modeling—A methodological approach from the finance industry," in *Proc. 1st Int. Work. Conf. Business Process Services Comput.*, Bonn, Germany, W. Abramowicz and L. A. Maciaszek, Eds., 2007, pp. 180–193.

[17] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering—A systematic literature review," *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, Jan. 2009.

[18] A. Brown and S. K. Johnston, "Using service-oriented architecture and component-based development to build web service applications," *Rational Softw. Corp.*, vol. 1, pp. 1–16, Jan. 2003.

[19] S. H. Chang and S. D. Kim, "A systematic approach to service-oriented analysis and design," in *Product-Focused Software Process Improvement* (Lecture Notes in Computer Science), vol. 4589. Berlin, Germany: Springer, 2007, pp. 374–388.

[20] C. M. MacKenzie, K. Laskey, F. McCabe, P. F. Brown, and R. Metz, *Reference Model for Service Oriented Architecture 1.0*, OASIS Standard 12(S18), 2006.

[21] M. Van Steen and A. S. Tanenbaum, "Distributed systems: Introduction," in *Distributed Systems*, 3rd ed., Leiden, The Netherlands: Maarten van Steen Leiden, 2018, p. 582.

[22] W. M. P. van der Aalst, "Process-aware information systems: Lessons to be learned from process mining," in *Transactions on Petri Nets and Other Models of Concurrency II: Special Issue on Concurrency in Process-Aware Information Systems*, K. Jensen and W. M. P. van der Aalst, Eds., Berlin, Germany: Springer, 2009, pp. 1–26.

[23] T. Erl, *SOA: Principles of Service Design*. Upper Saddle River, NJ, USA: Prentice-Hall, 2008.

[24] K. B. Laskey and K. Laskey, "Service oriented architecture," *Wiley Interdiscipl. Rev., Comput. Statist.*, vol. 1, no. 1, pp. 101–105, 2009.

[25] A. Erradi, S. Anand, and N. Kulkarni, "SOAF: An architectural framework for service definition and realization," in *Proc. IEEE Int. Conf. Services Comput. (SCC)*, Chicago, IL, USA, Sep. 2006, pp. 151–158.

[26] A. Nikaj, S. Mandal, C. Pautasso, and M. Weske, "From choreography diagrams to RESTful interactions," in *Service-Oriented Computing—ICSOC Workshops* (Lecture Notes in Computer Science), vol. 9586. Berlin, Germany: Springer, 2016, pp. 3–14, doi: 10.1007/978-3-662-50539-7_1.

[27] G. Blinowski, A. Ojdowska, and A. Przybylek, "Monolithic vs. microservice architecture: A performance and scalability evaluation," *IEEE Access*, vol. 10, pp. 20357–20374, 2022.

[28] A. Nikaj, M. Weske, and J. Mendling, "Semi-automatic derivation of RESTful choreographies from business process choreographies," *Softw. Syst. Model.*, vol. 18, no. 2, pp. 1195–1208, Apr. 2019.

[29] A. Arsanjani, S. Ghosh, A. Allam, T. Abdollah, S. Ganapathy, and K. Holley, "SOMA: A method for developing service-oriented solutions," *IBM Syst. J.*, vol. 47, no. 3, pp. 377–396, 2008.

[30] E. Souza and A. Moreira, "Deriving services from KAOS models," in *Proc. 33rd Annu. ACM Symp. Appl. Comput.*, New York, NY, USA, Apr. 2018, pp. 1308–1315.

[31] A. Arsanjani, "Service-oriented modeling and architecture," *IBM Developer Works*, vol. 1, pp. 1–15, Jan. 2004.

[32] S. Hekmat, S. Parsa, and B. Vaziri, "A new semi-automated method for service identification," *J. Web Eng.*, vol. 21, pp. 569–608, Feb. 2022.

[33] B. Andersson, P. Johannesson, and J. Zdravkovic, "Aligning goals and services through goal and business modelling," *Inf. Syst. e-Business Manage.*, vol. 7, no. 2, pp. 143–169, Mar. 2009.

[34] C. Ouyang, M. Dumas, W. M. P. V. D. Aalst, A. H. M. T. Hofstede, and J. Mendling, "From business process models to process-oriented software systems," *ACM Trans. Softw. Eng. Methodol.*, vol. 19, no. 1, pp. 1–37, Aug. 2009.

[35] *Business Process Model and Notation (BPMN) Version 2.0*, Object Management Group, HQ, Milford, MA, USA, 2011.

[36] *Unified Modeling Language (UML) Version 2.5*, Object Management Group, HQ, Milford, MA, USA, 2011.

[37] A. Scheer, O. Thomas, and O. Adam, "Process modeling using event-driven process chains," in *Process-Aware Information Systems*, M. Dumas, W. M. P. van der Aalst, and A. H. M. T. Hofstede, Eds., Hoboken, NJ, USA: Wiley, Sep. 2005, ch. 6, pp. 119–145.

[38] W. E. McCarthy, "The REA accounting model: A generalized framework for accounting systems in a shared data environment," *Accounting Rev.*, vol. 57, no. 3, pp. 554–578, 1982.

[39] G. L. Geerts and W. E. McCarthy, "An ontological analysis of the economic primitives of the extended-REA enterprise information architecture," *Int. J. Accounting Inf. Syst.*, vol. 3, no. 1, pp. 1–16, Mar. 2002.

[40] A. Osterwalder and Y. Pigneur, "An ontology for e-business models," in *Value Creation From E-Business Models*, W. L. Currie, Ed., Amsterdam, The Netherlands: Elsevier, 2004, ch. 4, pp. 65–97.

[41] J. Gordijn and H. Akkermans, "Designing and evaluating e-business models," *IEEE Intell. Syst.*, vol. 16, no. 4, pp. 11–17, Jul. 2001.

[42] ITU-T, *ITU-T, User Requirements Notation (URN)—Language Definition*, Int. Telecommun. Union (ITU) Geneva, Switzerland, 2012.

[43] A. Dardenne, A. van Lamsweerde, and S. Fickas, "Goal-directed requirements acquisition," *Sci. Comput. Program.*, vol. 20, nos. 1–2, pp. 3–50, Apr. 1993.

[44] S. C. Yu, *The Structure of Accounting Theory*. Gainesville, FL, USA: Univ. Press Florida, 1976.

[45] D. N. Boote and P. Beile, "Scholars before researchers: On the centrality of the dissertation literature review in research preparation," *Educ. Researcher*, vol. 34, no. 6, pp. 3–15, Aug. 2005.

[46] D. Birkmeier, S. Klöckner, and S. Overhage, "A survey of service identification approaches–classification framework, state of the art, and comparison," *Enterprise Model. Inf. Syst. Architectures (EMISA)*, vol. 4, no. 2, pp. 20–36, Dec. 2015.

[47] P. Brown, J. A. Estefan, K. Laskey, F. G. McCabe, and D. Thornton, "Reference architecture foundation for service oriented architecture version 1.0," Oasis Open, Tech. Rep., Jul. 2006. [Online]. Available: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm

[48] B. Bani-Ismail and Y. Baghdadi, "A survey of existing evaluation frameworks for service identification methods: Towards a comprehensive evaluation framework," in *Communications in Computer and Information Science*, vol. 877, L. Uden, B. Hadzima, and I.-H. Ting, Eds., Cham, Switzerland: Springer-Verlag, 2018, pp. 191–202.

[49] M. Templier and G. Paré, "Transparency in literature reviews: An assessment of reporting practices across review types and genres in top IS journals," *Eur. J. Inf. Syst.*, vol. 27, no. 5, pp. 503–550, Sep. 2018.
[50] *Mendeley Reference*, Elsevier, Amsterdam, The Netherlands, 2022.
[51] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *Proc. 18th Int. Conf. Eval. Assessment Softw. Eng.*, May 2014, pp. 1–10.
[52] M. P. Papazoglou and W.-J.-V. D. Heuvel, "Service-oriented design and development methodology," *Int. J. Web Eng. Technol.*, vol. 2, no. 4, pp. 412–442, 2006.
[53] *Service Oriented Architecture Modeling Language (SoaML) Version 1.0.1*, Object Manage. Group, HQ, Milford, MA, USA, 2012.
[54] P. Hruby, *Model-Driven Design Using Business Patterns*. Berlin, Germany: Springer, 2006.
[55] V. Kartseva, J. Gordijn, and Y.-H. Tan, "Designing value-based inter-organizational controls using patterns," in *Design Requirements Engineering: A Ten-Year Perspective* (Lecture Notes in Business Information Processing), W. van der Aalst, S. Ram, M. Rosemann, C. Szyperski, and G. Guizzardi, Eds., Berlin, Germany: Springer, 2009, pp. 276–301.
[56] W. M. P. van Der Aalst, A. H. M. Ter Hofstede, B. Kiepuszewski, and A. P. Barros, "Workflow patterns," *Distrib. Parallel Databases*, vol. 14, pp. 5–51, Jan. 2003.
[57] N. Russell, A. H. M. T. Hofstede, D. Edmond, and W. M. P. van der Aalst, "Workflow data patterns: Identification, representation and tool support," in *Conceptual Modeling* (Lecture Notes in Computer Science), vol. 3716, 1st ed., L. Delcambre, C. Kop, H. Mayr, J. Mylopoulos, and O. Pastor, Eds., Berlin, Germany: Springer, 2005, pp. 353–368.
[58] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, 1st ed., Cambridge, MA, USA: Addison-Wesley Professional, 1994.
[59] R. S. Huergo, P. F. Pires, F. C. Delicato, M. P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann, "MDCSIM: A method and a tool to identify services," *IT Converg. Pract.*, vol. 2, no. 4, pp. 1–27, 2014.
[60] H. Leopold, F. Pittke, and J. Mendling, "Automatic service derivation from business process model repositories via semantic technology," *J. Syst. Softw.*, vol. 108, pp. 134–147, Oct. 2015.
[61] A. Delgado, F. Ruiz, and I. G.-R. De Guzmán, "A reference model-driven architecture linking business processes and services," in *Proc. Annu. Hawaii Int. Conf. Syst. Sci.*, Waikoloa Village, HI, USA, T. Bui, Ed., 2018, pp. 4651–4660.
[62] M. Daghaghzadeh and S. M. Babamir, "A model driven and clustering method for service identification directed by metrics," *Softw., Pract. Exper.*, vol. 51, no. 2, pp. 449–484, Feb. 2021.
[63] Y. Kim and K.-G. Doh, "The service modeling process based on use case refactoring," in *Proc. 10th Int. Conf.*, Poznan, Poland. Berlin, Germany: Springer, Apr. 2007, pp. 108–120, doi: 10.1007/978-3-540-72035-5_9.
[64] S. Adam, N. Riegel, and J. Doerr, "Deriving software services from business processes of representative customer organizations," in *Proc. Int. Workshop Service-Oriented Comput., Consequences Eng. Requirements*, Barcelona, Spain, Sep. 2008, pp. 38–45.
[65] O. E. Adali, B. Ozkan, O. Turetken, and P. Grefen, "Identification of service platform requirements from value propositions: A service systems engineering method," in *IFIP Advances in Information and Communication Technology*, vol. 629, L. M. Camarinha-Matos, X. Boucher, and H. Afsarmanesh, Eds., Cham, Switzerland: Springer, 2021, pp. 311–322.
[66] E. Alirezaei and S. Parsa, "A hybrid syntactic and semantic approach to service identification in collaborative networks," *IFIP Adv. Inf. Commun. Technol.*, vol. 463, pp. 652–659, Jan. 2015.
[67] V. Alkkiomäki and K. Smolander, "Service elicitation method using applied qualitative research procedures," in *Advanced Design Approaches To Emerging Software Systems*, X. Liu and Y. Li, Eds., Paris, France: IGI Global, Jan. 2011, ch. 1, pp. 1–17.
[68] L. G. Azevedo, F. Santoro, F. Baião, T. Diirr, A. Souza, J. F. De Souza, and H. P. Sousa, "A method for bridging the gap between business process models and services," *iSys-Brazilian J. Inf. Syst.*, vol. 6, pp. 62–98, Dec. 2013.
[69] D. Bianchini, C. Cappiello, V. De Antonellis, and B. Pernici, "Service identification in interorganizational process design," *IEEE Trans. Services Comput.*, vol. 7, no. 2, pp. 265–278, Apr. 2014.
[70] D. Q. Birkmeier, A. Gehlert, S. Overhage, and S. Schlauderer, "Alignment of business and IT architectures in the German federal government: A systematic method to identify services from business processes," in *Proc. 46th Hawaii Int. Conf. Syst. Sci.*, Wailea, HI, USA, Jan. 2013, pp. 3848–3857.
[71] M. Daoud, A. E. Mezouari, N. Faci, D. Benslimane, Z. Maamar, and A. E. Fazziki, "Automatic microservices identification from a set of business processes," in *Communications in Computer and Information Science*, vol. 1207, M. Hamlich, L. Bellatreche, A. Mondal, and C. Ordonez, Eds., Cham, Switzerland: Springer, 2020, pp. 299–315.
[72] V. Dwivedi and N. Kulkarni, "A model driven service identification approach for process centric systems," in *Proc. IEEE Congr. Services II*, Beijing, China, Sep. 2008, pp. 65–72.
[73] B. Elvesæter, C. Carrez, P. Mohagheghi, A.-J. Berre, S. G. Johnsen, and A. Solberg, "Model-driven service engineering with SoaML," in *Service Engineering*. Berlin, Germany: Springer, 2011, pp. 25–54.
[74] J. Weller, W. Esswein, J. Stark, and M. Juhrisch, "Meet the challenge in service identification: A ratio-based approach," in *Proc. 13th Pacific Asia Conf. Inf. Syst., IT Services Global Environ.*, Hyderabad, Telangana, Jan. 2009, p. 52.
[75] M. Nußbaumer, T. Vogel, and A. Fuchsloch, "From cross-organizational business process to public services," in *Proc. 16th Americas Conf. Inf. Syst.*, vol. 5, Lima, Peru, M. Santana, J. N. Luftman, and A. S. Vinz, Eds., Aug. 2010, pp. 3649–3663.
[76] P. Jamshidi, S. Khoshnevis, R. Teimourzadegan, A. Nikravesh, and F. Shams, "Toward automatic transformation of enterprise business model to service model," in *Proc. ICSE Workshop Princ. Eng. Service Oriented Syst.*, Vancouver, BC, Canada, May 2009, pp. 70–74.
[77] A. Kazemi, H. Haghighi, and F. Shams, "ABSIM: An automated business service identification method," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 23, no. 9, pp. 1303–1342, Nov. 2013.
[78] K. Klose, R. Knackstedt, and D. Beverungen, "Identification of services-a stakeholder-based approach to soa development and its application in the area of production planning," in *Proc. 15th Eur. Conf. Inf. Syst.*, H. Osterle, J. Schelp, and R. Winter, Eds., St. Gallen, Switzerland, 2007, pp. 1802–1814.
[79] T. Kohlborn, A. Korthaus, T. Chan, and M. Rosemann, "Identification and analysis of business and software services—A consolidated approach," *IEEE Trans. Services Comput.*, vol. 2, no. 1, pp. 50–64, Jan. 2009.
[80] M. Mohamed, B. S. Mohamed, and M. El Amine Chergui, "A hybrid particle swarm optimization for service identification from business process," in *Proc. 2nd World Conf. Complex Syst. (WCCS)*, Agadir, Morocco, Nov. 2014, pp. 122–127.
[81] A. Nikravesh, F. Shams, S. Farokhi, and A. Ghaffari, "2PSIM: Two phase service identifying method," in *On the Move To Meaningful Internet Systems* (Lecture Notes in Computer Science), vol. 7045, R. Meersman, T. S. Dillon, P. Herrero, A. Kumar, M. Reichert, L. Qing, B. C. Ooi, E. Damiani, D. C. Schmidt, J. White, M. Hauswirth, P. Hitzler, and M. K. Mohania, Eds., Berlin, Germany: Springer, 2011, pp. 625–634.
[82] L. Rodriguez-Martinez, H. Duran-Limon, M. Mora, and F. Rodriguez, "SOCA-DSEM: A well-structured SOCA development systems engineering methodology," *Comput. Sci. Inf. Syst.*, vol. 16, no. 1, pp. 19–44, 2019.
[83] G. Scheithauer and B. Hardegen, "Requirements engineering for SOA services with BPMN 2.0—From analysis to specification," in *Business Process Model and Notation* (Lecture Notes in Business Information Processing), vol. 95, R. Dijkman, J. Hofstetter, and J. Koehler, Eds., Berlin, Germany: Springer, 2011, pp. 160–165.
[84] W.-J. Shiang, H. Rau, and Y.-H. Lin, "Service identification of a collaborative design workflow in a dynamically changing environment," in *Proc. Int. Conf. Netw., Sens. Control*, Okayama, Japan, Mar. 2009, pp. 685–690.
[85] H. M. Shirazi, N. Fareghzadeh, and A. Seyyedi, "A combinational approach to service identification in SOA," *J. Appl. Sci. Res.*, vol. 5, no. 10, pp. 1390–1397, 2009.
[86] A. Souza, C. Capelli, F. Santoro, L. G. Azevedo, J. C. S. D. P. Leite, and T. Batista, "Service identification in aspect-oriented business process models," in *Proc. IEEE 6th Int. Symp. Service Oriented Syst. (SOSE)*, Irvine, CA, USA, J. Z. Gao, X. Lu, M. Younas, and H. Zhu, Eds., Dec. 2011, pp. 164–174.
[87] S. Kim, M. Kim, and S. Park, "Service identification using goal and scenario in service oriented architecture," in *Proc. 15th Asia–Pacific Softw. Eng. Conf.*, Beijing, China, 2008, pp. 419–426.
[88] F. Vares, M. J. Amiri, and S. Parsa, "Towards a model-driven development of enterprise systems," in *Proc. Int. Symp. Comput. Sci. Softw. Eng. Conf. (CSSE)*, Shiraz, Iran, Oct. 2017, pp. 42–48.
[89] P. Weiss, "Service-based realization of business processes driven by control-flow patterns," in *Software Engineering Techniques* (Lecture Notes in Computer Science), vol. 4980, Z. Huzar, R. Koci, B. Meyer, B. Walter, and J. Zendulka, Eds., Berlin, Germany: Springer, 2011, pp. 91–102.

[90] R. Yousef, M. Odeh, D. Coward, and A. Sharieh, "BPAOntoSOA: A generic framework to derive software service oriented models from business process architectures," in *Proc. 2nd Int. Conf. Appl. Digit. Inf. Web Technol.*, London, U.K., Aug. 2009, pp. 50–55.
[91] A. T. Zadeh, M. Mukhtar, S. Sahran, and Z. Lotfi, "Automated service identification framework (ASIF)," *J. Theor. Appl. Inf. Technol.*, vol. 83, no. 3, pp. 451–464, 2016.
[92] I. Zafar, F. Azam, M. W. Anwar, B. Maqbool, W. H. Butt, and A. Nazir, "A novel framework to automatically generate executable web services from BPMN models," *IEEE Access*, vol. 7, pp. 93653–93677, 2019.
[93] J. Zdravkovic and T. Ilayperuma, "Designing consumer-aligned services using business value modelling," *Int. J. Organisational Design Eng.*, vol. 2, no. 3, p. 317, 2012.
[94] World Wide Web Consortium (W3C), *Web Ontology Language(OWL)*, Cambridge, MA, USA, 2004.
[95] S. Smirnov, M. Weidlich, J. Mendling, and M. Weske, "Action patterns in business process model repositories," *Comput. Ind.*, vol. 63, no. 2, pp. 98–111, Feb. 2012.
[96] *Information Technology: Business Operational View—Part 1: Operational Aspects of Open-EDI for Implementation*, ISO Standard ISO/IEC 15944-1, Int. Org. Standardization, Geneva, Switzerland, 2011.
[97] J. McDermid, "The role of formal methods in software development," *J. Inf. Technol.*, vol. 2, no. 3, pp. 124–134, Sep. 1987.
[98] T. J. Ross, "Logic and fuzzy systems," in *Fuzzy Logic With Engineering Applications*, 3rd ed., Chichester, U.K.: Wiley, Jan. 2010, ch. 5, pp. 117–173.
[99] K. Alizadeh, M. A. Seyyedi, and M. Mohsenzadeh, "Mapping service concept and enterprise ontology in service identification," in *Proc. 7th Int. Conf. Networked Comput.*, Gumi, Korea (South), Sep. 2011, pp. 22–27.
[100] A. Erradi, N. Kulkarni, and P. Maheshwari, "Service design process for reusable services: Financial services case study," in *Service-Oriented Computing—ICSOC* (Lecture Notes in Computer Science), vol. 4749, B. Krämer, K. Lin, and P. Narasimhan, Eds., Berlin, Germany: Springer, 2007, pp. 606–617.
[101] M. Abdellatif, A. Shatnawi, H. Mili, N. Moha, G. E. Boussaidi, G. Hecht, J. Privat, and Y.-G. Guéhéneuc, "A taxonomy of service identification approaches for legacy software systems modernization," *J. Syst. Softw.*, vol. 173, Mar. 2021, Art. no. 110868.
[102] C. E. Salgado, J. Teixeira, N. Santos, R. J. Machado, and R. S. P. Maciel, "A SoaML approach for derivation of a process-oriented logical architecture from use cases," in *Exploring Services Science* (Lecture Notes in Business Information Processing), vol. 201, H. Nóvoa and M. Dragoicea, Eds., Cham, Switzerland: Springer, 2015, pp. 80–94.

REDOUANE BLAL received the M.Sc. degree in information technology from École des sciences de la gestion, in 2018. He is currently pursuing the Ph.D. degree specializing in information systems with the Management Ph.D. Program, Université du Québec à Montréal, QC, Canada.

He joined the Ph.D. program after spending over 20 years managing and designing IT/IS solutions for large organizations in different industries, including finance, trading, and the public sector. His recent works are focused on SOA service identification and modeling from business process models. His research interests include business process management (BPM), enterprise architecture, and model-driven architecture.

ABDERRAHMANE LESHOB received the Ph.D. degree in computer science from the University of Quebec in Montreal, Canada, in 2013.

After working for more than 20 years in the industry as a Senior Solutions and Enterprise Architect. He is currently an Associate Professor of information systems with the Department of Analytics, Operations, and Information Technology, University of Quebec in Montreal. His research and teaching interests include robotic process automation, business process management, enterprise architecture, software design, and model-driven engineering.

HAFEDH MILI received the Ph.D. degree in computer science (artificial intelligence) from George Washington University, in 1988.

He is currently a Professor with Université du Québec à Montréal. His research interests include software engineering, from the early phases of software development (business modeling, representation and classification of business processes, and generation of software models from business models) to the reengineering of legacy applications to adapt them to modern deployment architectures. He has consulted widely on business rules and co-authored *Agile Business Rule Development* (J. Boyer, Springer, 2011).

IMEN BENZARTI received the Ph.D. degree in computer science from Université du Québec à Montréal, Canada, in 2020.

She was a Postdoctoral Fellow with Université du Québec à Montréal, in 2021. She is currently a Professor of computer science with École de Technologie Supérieure, Canada. She is a member with the LATECE Research Laboratory, Université du Québec à Montréal. Her research interests include models-driven engineering, human-centric software engineering, and empirical software engineering.

PIERRE HADAYA received the bachelor's degree in electrical engineering and the Ph.D. degree in technology management from the École Polytechnique de Montréal, QC, Canada.

He is currently a Full Professor with the School of Management, Université du Québec à Montréal, QC, where he is the Co-Director of the Center of Expertise in Digital Transformation of Organizations (CETNO). As the Co-Founder of the ASATE Group, he also collaborates with organizations that want to transform themselves in order to develop a competitive advantage. His research and teaching activities focus on strategic management, organizational transformation and its governance, the business architecture approach, and the strategic alignment of IT systems within organizations (including IT Enterprise Architecture).

Mr. Hadaya is a member of the Board of Directors of the Association for Strategy Professionals as well the Editor-in-Chief of *Strategy Magazine*. He is also the Certified Corporate Director, having completed the university certification program in corporate governance of the Collège des Administrateurs de Sociétés.

RAQEEBIR RAB received the B.Sc. degree in science with a major in computer science from the Augustana Faculty, University of Alberta, Canada, in 2004, and the M.Sc. degree in computer science from Concordia University, Montreal, Canada, in 2012.

She is currently an Assistant Professor with the Department of Computer Science and Engineering (CSE), Ahsanullah University of Science and Technology (AUST), Dhaka, Bangladesh. Her research interests include wireless multihop networks (ad hoc and sensor networks) with an emphasis on mathematical modeling, performance analysis, protocol design, and data science.