Software and Systems Modeling (2023) 22:1857–1895
https://doi.org/10.1007/s10270-023-01109-1

SPECIAL SECTION PAPER

Design of blockchain-based applications using model-driven engineering and low-code/no-code platforms: a structured literature review

Simon Curty[1] · Felix Härer[1] · Hans-Georg Fill[1]

Received: 28 October 2022 / Revised: 25 April 2023 / Accepted: 2 May 2023 / Published online: 11 June 2023

Abstract
The creation of blockchain-based software applications requires today considerable technical knowledge, particularly in software design and programming. This is regarded as a major barrier in adopting this technology in business and making it accessible to a wider audience. As a solution, low-code and no-code approaches have been proposed that require only little or no programming knowledge for creating full-fledged software applications. In this paper we extend a review of academic approaches from the discipline of model-driven engineering as well as industrial low-code and no-code development platforms for blockchains. This includes a content-based, computational analysis of relevant academic papers and the derivation of major topics. In addition, the topics were manually evaluated and refined. Based on these analyses we discuss the spectrum of approaches in this field and derive opportunities for further research.

Keywords Blockchain · Low-code · No-code · Model-driven engineering · Software development

1 Introduction

With the further maturing of blockchain technologies and the on-going transition to more energy-efficient and faster protocols with higher transaction volumes [86, 189], a more widespread adoption of these technologies seems within reach. However, one considerable barrier limiting the adoption is the technical and organizational complexity that users are confronted with when creating blockchain-based applications [131]. This complexity originates on the one hand from the underlying technical foundations, which build on distributed and decentralized systems, cryptography, and algorithmic processing [12]. Blockchains such as Ethereum [272] combine these properties for storing transactions in an append-only data structure, where each new block has a cryptographically verifiable link to its predecessor. Thus, users are part of a decentralized network that minimizes the degree of trust required towards other participants who continuously validate the information of the blockchain [92, 93]. In addition, organizational barriers such as the involvement of new regulatory requirements, the development of new skills and competencies, and the availability of financial and human resources may prevent adoption in practice [56].

From the perspective of software engineering, the lack of specialists for programming may today be partly compensated with so-called *low-code* platforms [36, 95, 245]. These development platforms are typically available as cloud services with visual, diagrammatic interfaces and declarative languages. In our view, they constitute the next step in the industry adoption of academic model-driven engineering (MDE) approaches and their predecessors.

The long-standing MDE discipline originated from the development of CASE (computer-aided software engineering) tools in the late 1980s [171], proposing the use of models as primary development artifacts for engineering [42, 76, 266]. MDE methods and tools address requirements, architectural components, and software artifacts with professional developers.

Communicated by Iris Reinhartz-Berger and Dominik Bork.

✉ Simon Curty
simon.curty@unifr.ch

Felix Härer
felix.haerer@unifr.ch

Hans-Georg Fill
hans-georg.fill@unifr.ch

[1] Digitalization and Information Systems Group, University of Fribourg, Bd de Pérolles 90, 1700 Fribourg, Switzerland

The term *low-code development* has been adopted by industry in recent years where various platforms are provided for the generation of software components or their cloud-based integration. While *low-code* approaches allow a user to produce results without having to understand source code and there may be an underlying model integrated with features of the platform [36], the model may not conform to an explicit formalization [76]. Further, we consider so-called *no-code* approaches as a subset of low-code approaches that operate at an abstraction level above code, not showing code to the user at all. Today, a large number of such platforms and tools are available that either support the development of complete software applications or focus on providing specific functionality, e.g., for entering data in a form and saving it to a database [213].

For easing the creation of blockchain-based applications it seems obvious to revert to MDE and low-code approaches. These carry the potential to abstract from the technical complexity and enable users to focus on usage scenarios and the organizational embedding [94].

In this paper we present an extension of a prior study [67] which investigated academic and industrial approaches for realizing blockchain-based applications using these methods and provide a comparative analysis of model-driven and low-code methods in this area. The extension encompasses the literature review, the review of industrial approaches, and their comparison. The literature review is extended in its scope and time frame as well as methodology, in particular through the addition of a computational literature analysis with formalized qualitative assessments. The review of low-code and no-code approaches incorporates platforms until October 2022 with an additional analysis step adding application characteristics and further platform descriptions. The review sections are concluded by discussing and comparing academic approaches in relation to low-code and no-code platforms.

We will do this along the following four research questions:

RQ1: Which academic modeling approaches have been developed until today for designing blockchain-based applications?

RQ2: Which low-code and no-code platforms permit the realization of blockchain-based applications?

RQ3: What are the predominant characteristics and areas of academic modeling approaches as well as low-code and no-code platforms?

RQ4: What are future research opportunities not realized today by academic approaches and low-code or no-code platforms?

In particular, we will regard approaches that are already available for creating blockchain-based software applications or offer interfaces to other platforms enabling this. This will permit to describe the state-of-the-art in this area and derive requirements for the development of future approaches. The remainder of the paper is structured as follows. Section 2 will outline foundations and related work in the form of previous studies and lead over to the description of our research methodology in Sect. 3. Subsequently, we will present in Sect. 4 our review of academic MDE approaches and in Sect. 5 the review of low-code and no-code development platforms offered by industry. Section 6 discusses the results and insights gained through our analysis and leads to our conclusion in Sect. 7.

2 Foundations and related work

In the following we briefly describe the main characteristics of blockchains and smart contracts as an introduction for readers who are not familiar with these technologies. Subsequently, we will outline the related surveys on model-driven approaches for the design of blockchain-based applications.

2.1 Blockchains and smart contracts

Blockchains are a family of technologies where transactions between authorized parties are stored in an electronic distributed ledger in a decentralized, distributed, immutable, and transparent way [93]. This is achieved through so-called consensus algorithms that guarantee the validity and integrity of transactions. The term "blockchain" stems from the form of the data structure of the ledger: a cryptographically linked chain of blocks containing records of transactions. The access to the ledger may either be restricted to certain parties (permissioned blockchain) or it may be openly accessible (public blockchain).

Blockchain-based applications then rely on this technology for its security and integrity properties, as financial channel, or to benefit from an established distributed infrastructure, e.g., by using a public blockchain. Furthermore, some blockchains allow applications to employ so-called *smart contracts*, pieces of code added to transactions for the decentralized execution of algorithms [12].

2.2 Related surveys

Developing blockchain-based applications requires a high level of expertise and understanding of the underlying technologies. Blockchain-based applications are empowered by smart contracts, i.e., programs executed on the blockchain. These smart contracts often involve financial transactions or deal with issues related to trust. As such, their correctness is of utmost importance. Due to the immutable nature of blockchains, mistakes in smart contract implementations are difficult to rectify. Many works from different fields in computer science investigate these issues and numerous surveys

Table 1 Selected surveys on (i) smart contract formalization, verification, engineering, and languages; (ii) blockchain-based business process management, and (iii) MDE techniques for the development of blockchain applications. These surveys have been considered in the review process as well.

References	Title
Ait Hsain et al. [3]	Ethereum's smart contracts construction and development using model driven engineering technologies: a review
Alam et al. [4]	Blockchain domain-specific languages: survey, classification, and comparison
Almakhour et al. [6]	Verification of smart contracts: a survey
Dwivedi et al. [81]	Legally enforceable smart-contract languages: a systematic literature review
Fahmideh et al. [85]	Engineering blockchain based software systems: foundations, survey, and future directions
García-García et al. [103]	Using blockchain to improve collaborative business process management: systematic literature review
Grishchenko et al. [108]	Foundations and tools for the static analysis of ethereum smart contracts
Härer and Fill [122]	A comparison of approaches for visualizing blockchains and smart contracts
Hu et al. [133]	A comprehensive survey on smart contract construction and execution: paradigms, tools, and systems
Levasseur et al. [152]	Survey of model-driven engineering techniques for blockchain-based applications
Liu et al. [157]	A survey on security verification of blockchain smart contracts
Pinna et al. [202]	On the use of petri nets in smart contracts modeling, generation and verification
Sánchez-Gómez et al. [215]	Model-based software design and testing in blockchain smart contracts: a systematic literature review
Singh et al. [224]	Blockchain smart contracts formalization: approaches and challenges to address vulnerabilities
Tolmach et al. [246]	A survey of smart contract formal specification and verification
Udokwu et al. [254]	Evaluation of approaches for designing and developing decentralized applications on blockchain
Varela-Vaca et al. [257]	Smart contract languages: a multivocal mapping study

were conducted from different perspectives. Table 1 provides an overview of some related surveys, collected during the review process as elaborated in Sect. 4.1.

Most surveys focus on issues related to the development and analysis of smart contracts, while few regard blockchain applications as a broader topic. A general survey on various aspects of smart contract development and execution is presented in [133]. This includes a review of tools and techniques for the analysis and verification of smart contracts, e.g., to detect vulnerabilities, design patterns, smart contract languages, and execution schemes. While various issues related to software engineering are discussed, model-driven or low-code techniques to develop blockchain-based software are not regarded. The surveys [4, 81, 122, 257] review smart contract languages, including some of the work presented in this paper, but do not focus on MDE or low-code approaches specifically. While visual programming languages aim to reduce complexity and improve accessibility for the programmer, they do not correspond in general to low-code development approaches, which may involve visual programming but also deal with the generation and life-cycle management of software artifacts.

Blockchain-based approaches in the field of business process management have been reviewed in [103] with a particular focus on collaborations. While approaches from this field are oftentimes not directly concerned with MDE, the model-based execution of business processes, workflows and choreographies is a central issue.

Formal specification and verification techniques for smart contracts is an actively researched field with numerous surveys [6, 108, 157, 202, 224]. This field is concerned with the detection or prevention of security vulnerabilities and faults, code optimization, formalization of semantics, and the correctness of smart contract implementations. Some approaches of this field employ MDE techniques, e.g., the generation of code from models.

A general survey on software engineering approaches for blockchain applications is presented in [85], including some that apply MDE techniques. However, a comprehensive overview of MDE is not in the scope of this former survey. The surveys [3, 152, 215, 254] review MDE approaches in particular. In [3] the authors discuss MDE specifically for Ethereum smart contracts, however, the review process is not elaborated. Sánchez-Gómez et al. [215] review model-based testing and development approaches and Udokuwu

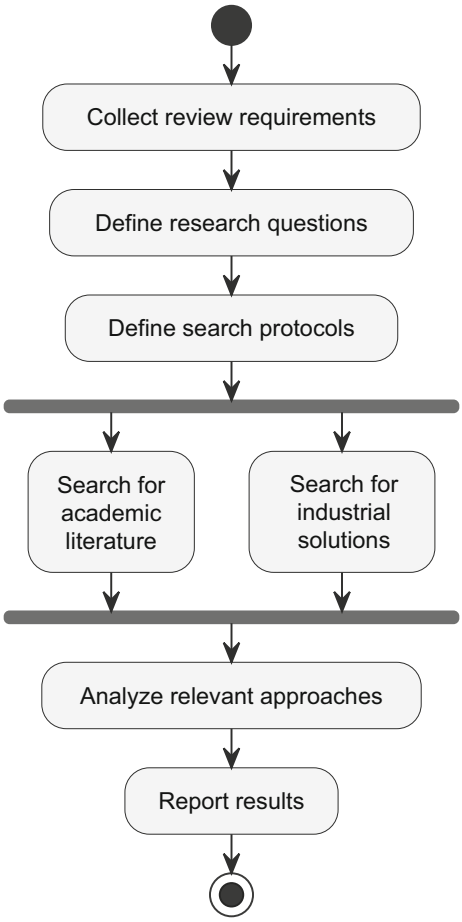

Fig. 1 Overall process for this study. Since the data sources for academic and industrial approaches differ significantly, two separate search protocols were formulated. The protocols were executed in parallel and isolated

et al. [254] focus on the evaluation of software development processes. Since the publication of these surveys in 2020, newer approaches have emerged. A more recent review of MDE methods was conducted by Levasseur et al. [152] in 2021. In comparison to their work, we applied a broader search methodology and identified more approaches. None of these studies considers industrial approaches such as low-code platforms and they focus predominantly on blockchain-based application development.

In summary, while numerous studies on issues regarding smart contract development have been conducted, to the best of our knowledge, a comprehensive review of the state-of-the-art of MDE and low-code approaches from both academia and industry in this field is missing so far.

3 Research methodology

For answering the four research questions we will employ the following research methodology. At first, we review existing academic MDE approaches for blockchain applications in the form of a structured literature review (SLR), followed by a content-based computational analysis of the resulting full text documents using topic modeling [54] and a review of state-of-the-art industrial low-code and no-code software platforms (Fig. 1).

The review of MDE approaches is two-fold. Relevant papers were manually retrieved through the structured literature review at first, including the identification of categories and topics. In addition, topics were then identified using a content-based computational analysis for the resulting papers based on their full texts. Both results were finally combined in a manual rating and assessment process. In this way, the validation of the manually compiled and computational results is ensured. In particular, the SLR involved manual retrieval and paper screenings by the authors for identifying relevant papers and topics. Regarding the process, we follow the guidelines by Webster and Watson [264] and vom Brocke et al. [260]. The initial corpus of the SLR was generated by searching all keyword combinations from two groups, where group one included '*blockchain, distributed ledger, smart contract*' and group two '*enterprise model, conceptual model, business model, model-driven, no-code, low-code*'. These keywords were selected based on the domain understanding of the authors. We expected the relevant concepts to be dispersed, thus we chose a broad set of keywords.

Subsequently, the topics were identified in a computational analysis in multiple iterations by applying Latent Dirichlet Allocation (LDA) [54]. The final synthesis of the resulting papers and their topics follows the processes applied by Casalaro et al. [49] for the synthesis and Torres et al. [49, 248] regarding the rating and qualitative assessment through inter-rater reliability measures.

For discovering relevant industrial low-code and no-code software platforms, we reverted further to expert knowledge from industry in the field of low-code development combined with our own searches. On this basis, we conducted a survey of available platforms towards suitability for blockchain application development, and identified according characteristics, application areas, and categories. We discuss platforms in each category together with representative examples for assessing the state-of-the-art in implemented industry software solutions. This exploratory research approach is directed towards discovering requirements for future platforms that combine blockchain application development with the state-of-the-art from academia and industry.

4 Academic MDE approaches

In the following subsections, we review approaches of the academic discipline *model-driven engineering* in regard to development solutions for blockchains.

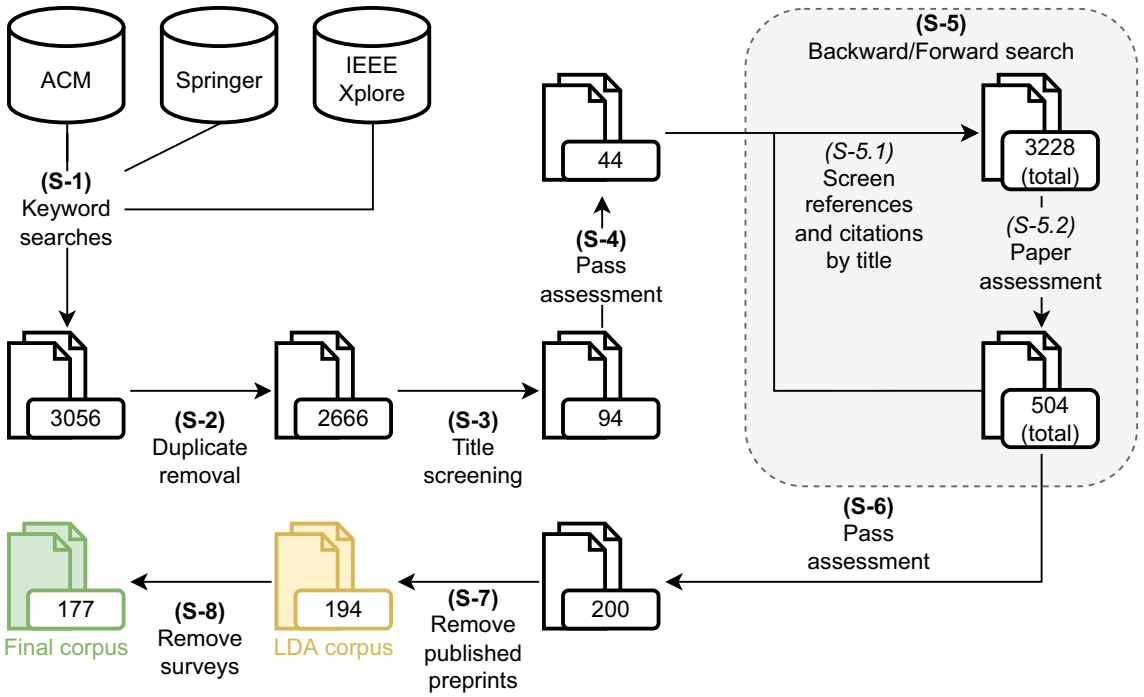

Fig. 2 Systematic review process for the identification of academic MDE-related documents. A keyword search was performed on the portals ACM, IEEE Xplore, and Springer. This corpus was filtered by title and then subject to an assessment regarding the documents' full-texts. Subsequently, a recursive backward-forward search was performed, resulting in a set of 200 relevant documents. Further, any surveys and preprints that have a published version were removed. This final set was then used for a content-based analysis using LDA

Model-driven engineering introduces models as primary artifact to the software development process in order to address numerous challenges of software engineering [42, 218]: First, the common understanding of software artifacts can be facilitated by domain-specific models, as such models are easier to interpret for humans than code. Second, model-based reasoning allows the verification of software, e.g., to determine the fulfillment of security properties. And third, well-defined models allow developers to create software artifacts in an automated fashion, which are correct-by-construction, with no or reduced coding effort. To identify existing MDE approaches that target specifically the development of blockchain applications, we conducted a systematic literature review as elaborated in the following.

4.1 Review process

The systematic review process as shown in Fig. 2 follows the guidelines by Webster and Watson [264] and vom Brocke et al. [260]. To obtain an initial corpus of documents, we performed keyword searches in step (S-1) on ACM, Springer, and IEEE Xplore with the search strings shown in Table 2. These publishers were chosen as starting point as we had access to the full-texts on their platforms. Moreover, established outlets of the modeling domain are mostly published on these platforms [123]. The steps (S-5.1) and (S-5.2) ensured that outlets from other publishers are considered as well. In total, the review covers 2173 outlets—a wide array of conferences, consortia, forums, journals, magazines, symposia, and workshops. The search strings are informed by previous studies in this area, namely [67, 123]. In the pre-study [67] we had found that restricting the search criteria on terms closely related to model-driven and low-code approaches does not cover fields we deem relevant nonetheless. This is oftentimes due to differences in the terminology used. Thus, we widened the search with the addition of terms related to models of a specific nature. The study in [123] had shown that the precise meaning assigned to the terms "modeling" or "model" is contextual and varies across fields of study. We thus restricted the search terms to account for the broad use of these keywords. Thereby, the term "business modeling" is largely motivated by approaches in the field of value and goal modeling, the term "conceptual modeling" by domain models and ontologies used during any stage of software development, and lastly the term "enterprise modeling" by the field of enterprise architectures.

From this initial corpus of documents, duplicates were removed in step (S-2). Before the full-text analysis, the reduced corpus was then screened by titles in step (S-3). As basis for this third step we formulated loose keyword

Table 2 The five search strings (in a simplified form) used for retrieving the initial corpus, and the number of results found on ACM, IEEE Xplore, and Springer. The exact syntax of search strings varies for each search portal. The search period for academic approaches was restricted to publications between 2014 and July 2022

Search string	# results
("blockchain" OR "distributed ledger" OR "smart contract" OR "smart-contract") AND ("business model" OR "business modeling") AND (year > 2014)	2150
("blockchain" OR "distributed ledger" OR "smart contract" OR "smart-contract") AND ("conceptual model" OR "conceptual modeling") AND (year > 2014)	507
("blockchain" OR "distributed ledger" OR "smart contract" OR "smart-contract") AND ("model driven" OR "model-driven") AND (year > 2014)	235
("blockchain" OR "distributed ledger" OR "smart contract" OR "smart-contract") AND ("no code" OR "no-code" OR "low code" OR "low-code") AND (year > 2014)	114
("blockchain" OR "distributed ledger" OR "smart contract" OR "smart-contract") AND ("enterprise model" OR "enterprise modeling") AND (year > 2014)	50

criteria, whereby the title should contain a term related to a model or the activity of modeling, and mention a blockchain-related word, such as "*distributed*", "*chain*", or "*contract*". Non-exhaustive examples of satisfying the former criteria include the mentioning of modeling languages, model-driven practices, conceptualizations, structured knowledge representations, and domains known for applying MDE practices. Excluded were documents with titles indicating the discussion of a business model for a particular use case, since these occurred in abundance but do not in general relate to MDE. The title screening in step (S-3) was performed by all authors in isolation, whereby each author noted whether or not a publication seemed relevant and refined the assessment in a second pass. For documents without full agreement by all authors, consensus was established through a discussion.

In the next step, the documents were assessed by at least reading the abstract and reviewing tables and images (S-4), considering the inclusion criteria that (i) the documents should be related to distributed ledger technologies, and (ii) create, discuss, or present a modeling activity. Publications using models to only illustrate software, systems, or a use case, e.g., by means of a standard UML use case diagram, were excluded. For documents that passed the aforementioned assessment we then extracted dimensions for comparison by reading the full-texts. The dimensions are presented in 4.2.

For these 44 documents, we then performed a recursive backward-forward search (S-5), as proposed in [262]: references and citations were screened by title, relevant publications added to the set (S-5.1), and subsequently assessed (S-5.2). Step (S-5.2) was performed in the same manner as step (S-4). Surveys and versions of already captured documents, namely preprints or published versions thereof, were first considered as well. For all relevant new additions, a backward-forward search was again performed. Eventually, no new relevant documents could be found, and the backward-forward search was concluded. Of all thus collected documents, 200 fulfilled the assessment of being in the scope of the literature study by reading the full-texts of the papers (S-6). We further removed preprints for which a published version is contained in the corpus (S-7). The resulting corpus served as basis for a content-based analysis using LDA. Finally, surveys were removed in step (S-8), resulting in the set of documents presented in the following. This was done since these surveys were considered secondary studies and do not present an original approach, but rather focus on the synthesis of previous approaches.

4.2 Results of the literature review process

Through the SLR we identified 177 academic publications relevant in the context of MDE for blockchain-based applications. A discussion of the academic approaches is presented in Sect. 6. Each publication was subject to a classification based on three dimensions. The choice of these particular dimensions is informed by a pre-study [67].

Base language: This dimension refers to the modeling language that was applied in the approach or used as foundation for an extension, for example, in the form of a profile or through an addition of domain-specific concepts. Some approaches do not revert to such a base language. Instead, they propose a tailor-made method for modeling some aspect of a blockchain system. We therefore refer to any such method as a *domain-specific language (dsl)*. This domain-specificity is to be understood in the context of the development and modeling of blockchain-based applica-

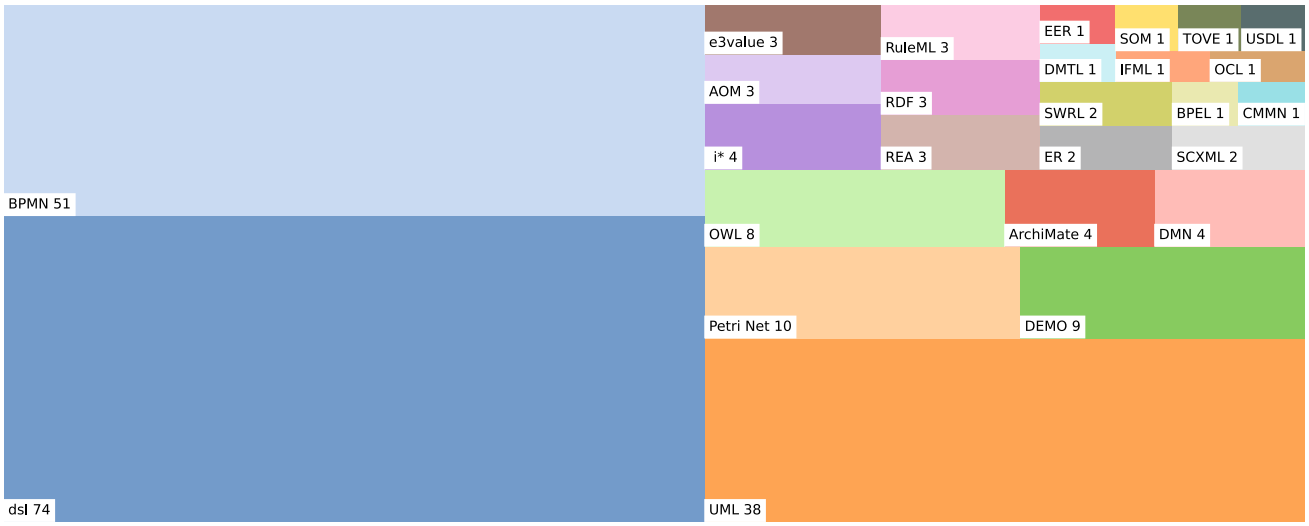

Fig. 3 Treemap of the identified modeling languages, that have been employed by academic approaches—approaches may use or propose multiple languages, resulting in a total of 232 assignments. The area of a rectangle represents the number of approaches using this language: domain-specific language (*dsl*, 74), BPMN (51), UML (38), Petri Net (10), DEMO (9), OWL (8), ArchiMate (4), DMN (4), i* (4), REA (3), AOM (3), RDF (3), e^3 value (3) RuleML (3), ER (2), SCXML (2), SWRL (2), BPEL (1), CMMN (1), DatalogMTL (DMTL, 1), IFML (1), OCL (1), EER (1), SOM (1), TOVE (1), and USDL (1)

tions. Furthermore, we did not distinguish between different versions of languages.

Blockchain technology: This dimension is defined as the base blockchain technology relevant for the approach. That is, the technology that the approach either applies, analyzes or, targets. For approaches of a *conceptual* nature that do not specify a technology, or for which considering such specifics is not relevant, we denote the blockchain technology to be *unspecified*. Some approaches target multiple blockchain systems, for which we assign the value *multiple*. The use of programming languages compiled to bytecode for Ethereum's virtual machine [272] is regarded as using the Ethereum blockchain technology, regardless of the existence of other blockchains compatible with such bytecode. An implementation of non-established blockchain systems is denoted as *custom*. We did not distinguish between developments under an overarching umbrella, namely the various Hyperledger projects. Furthermore, the distinction between permissioned and permissionless networks is not considered, since this is generally an issue of system and network configuration.

Nature of realization: We distinguish between *executable code generating* and *conceptual* approaches. The former refers to an approach that automatically or partially automatically generates code artifacts from some model representation. These artifacts must be executable using a blockchain technology. Such artifacts include but are not limited to smart contracts, blockchain connectors, integration facilities, user-interface components for blockchain applications and process or workflow encodings.

We denote those approaches as *conceptual* that do not satisfy the code generation criterion, but instead focus on the conceptualization of the blockchain domain, resulting in a model representation, reasoning on blockchain systems based on models, conceptual language mappings or model-based methodologies for the design and development of blockchain applications. Conceptual artifacts produced by such approaches include but are not limited to ontologies, metamodels, methodologies, semantic models, and business models.

4.2.1 Employed modeling languages

The academic approaches at hand may use or propose one or several modeling languages. These have been identified according to the previously defined base language dimension. In this section we present the modeling languages ordered by the number of occurrences (totaling in $n = 232$) in terms of the number of applications in academic approaches (c.f. Fig. 3). In total, we have identified 25 base languages, most of which are only sparingly used for the development of blockchain-based applications. In the following, the languages and their primary function in the context of DLT are briefly summarized, while a discussion of the approaches will follow in Sect. 6.

Domain-specific language (dsl): Papers proposing or applying a domain-specific language for the blockchain domain form the largest group with 74 representatives (out of 232). This group is notably heterogeneous, ranging from declarative (e.g., [125, 149, 207]) to graphical (e.g., [31, 39,

265]) modeling approaches with a varying degree of formalization. Some approaches combine a *dsl* with non-blockchain specific modeling languages, for example for the model-based generation of smart contract code (e.g., [128, 256]).

Business process model and notation (BPMN) [238]: With 51 occurrences BPMN is the most widely employed standard modeling language in the corpus of documents. In the context of blockchain-based applications, BPMN has been applied, for example, to execute business processes on-chain, based on smart contracts (e.g., [159]), or for the collaborative execution and enforcement of choreographies (e.g., [146]).

Unified modeling language (UML) [241]: UML is the second most prominent modeling language after BPMN with 38 representatives. This group accounts for all types of UML diagrams, although UML class diagrams are the most widely used by the approaches at hand. Approaches reverting to UML include development methodologies (e.g., [168]), conceptual modeling and ontology design (e.g., [70, 194]), as well as model-based code generation (e.g., [113]).

Petri Nets [199, 200]: Petri Nets ($n = 10$) allow for the description of concurrent processes in distributed systems. Thereby, system states and transitions between these are modeled with a formalized diagrammatic notation. Applications of Petri Nets in the context of blockchain technology include, for example, the formalization of ontologies (e.g., [78]), smart contract simulation and verification (e.g., [277]) and the encoding of business processes for execution engines (e.g., [102]).

Design and engineering methodology for organizations (DEMO) [74, 75]: DEMO ($n = 9$) is a methodology with an ontological foundation for the design (and re-design) of organizations. A main concern of DEMO is the separation of the abstract representation of an organization's operation from its realization [74]. These essential features are described by means of four diagrammatic models. The approaches at hand apply DEMO, or parts thereof, for various purposes. For example, the generation of code (e.g., [226]), and the use as ontological foundation (e.g., [70]). Of particular interest as well is the actor transaction diagram of DEMO (e.g., [223]).

Web ontology language (OWL) [261]: OWL ($n = 8$) is an ontology language for the semantic web with formally defined semantics. An OWL ontology may be represented in various ways, for example, as an RDF graph (see description below), which is the main exchange syntax of an OWL document, or with an UML class diagram. Several academic approaches express DLT-related domain models in OWL (e.g., [26, 28]). The formalized semantics of OWL allow for the application of formal reasoning mechanisms in order to gain new insights, in particular in combination with query and rule languages (e.g., [29]).

ArchiMate [151, 244]: ArchiMate ($n = 4$) is a multi-layered enterprise architecture modeling language. The layers represent various aspects of an enterprise, ranging from strategy and business concerns to the technical infrastructure. Academic approaches employ ArchiMate to model blockchain systems and the interrelations with business models (e.g., [66, 135]), for the creation of reference models (e.g., [82]), and as input for the generation of software artifacts based on a concept mapping (e.g., [19]). The application and technology layers of ArchiMate are considered, for example, in [66], while other approaches may regard mainly the top-level layers (e.g., strategy [135] or business layer [82]).

Decision model and notation (DMN) [243]: DMN ($n = 4$) is a business decision modeling language, designed to be usable as complementary modeling method alongside BPMN, thereby allowing the decisions to be integrated in business processes. The combination of DMN and blockchain technologies was explored for example for the purpose of executing decision models on-chain, as seen in [115–117], or for supporting decision making as part of a development methodology [192].

i* strategic actor relationships modeling framework (i*) [274–276]: The i* framework ($n = 4$) is a graphical modeling method where information systems are regarded as socio-technical systems in which actors are related to another in regards to goals, tasks and resources. This framework has been discussed, e.g., for requirements engineering and goal modeling for supporting the development of decentralized applications in an organizational setting [118, 258, 259], and for the policy compliant generation of smart contract code [252].

Agent-oriented Modeling (AOM) [230]: AOM ($n = 3$) is a methodology for modeling socio-technical systems. Thereby, an information system is represented by interactions among autonomous agents. Several works regard AOM as integral part of some development methodology for decentralized applications, e.g., [153, 253, 255].

e^3value [107]: The e^3 value language ($n = 3$) provides a notation to model value streams between actors. It is thus mostly used to explore business models and use cases with a focus on value exchanges. The usage in the context of blockchains is two-fold: either employed as part of a smart contract development methodology (e.g., [105]) or for use case modeling and analysis. In the latter case, the main concern is, e.g., the identification of DLT use cases [203] or the modeling thereof [198].

Resource–Event–Agent (REA) [175]: The REA model ($n = 3$) provides a general business ontology and modeling notation for expressing economic resources, economic events, and economic agents and their relations. It has initially been proposed as an accounting framework for shared data environments with a data modeling notation resembling the Entity-Relationship model. The business ontology

of REA is used in combination with DEMO, e.g., in [69, 70] for the development of blockchain domain ontologies.

Resource description framework (RDF) [124]: The RDF standard ($n = 3$) provides a notation for the exchange and description of graph data. Thereby, a graph is represented as collection of triple statements. While RDF was originally intended as description language for metadata in the semantic web, it has found more general use since then, e.g., for modeling and exchanging ontologies. For example, blockchain-related ontologies have been expressed in RDF, as seen in [10, 201, 220].

Rule markup and modeling language (RuleML) [212]: RuleML ($n = 3$) is a standard for the semantic web based on RDF and XML for the specification of rules and constraints regarding the transformation and semantic interpretation of data. Several other markup languages have been developed based on RuleML, e.g., SWRL. Approaches at hand apply RuleML, for example, for the verification of DApps in regard to compliance rules [256]. An extended version of RuleML is proposed as a declarative smart contract language [71, 72].

Entity-relationship model (ER) [52]: The ER model ($n = 2$) has originally been proposed for the design of database models, but is commonly used for general data modeling tasks as well. It has been suggested to revert to ER for the design of the data domain model as part of the blockchain application development methods presented in [100, 208].

State chart extensible markup language (SCXML) [25]: SCXML ($n = 2$) is a W3C standard for the declaration of state machines in an XML format. While numerous approaches at hand involve state machines, only two revert to this standard [185, 216]. In these works, SCXML is used as input format for a blockchain-based engine, with the purpose of executing smart contracts modeled as state machines.

Semantic web rule language (SWRL) [132]: The W3C standard SWRL ($n = 2$) is a combination of OWL and RuleML, expanding the expressiveness of OWL with the capability of formulating constraints or rules on ontological concepts. Together with ontologies formalized in OWL, SWRL enables deductive reasoning and knowledge inference. This has been used, e.g., in [29] for gaining insights in DApps modeled according to an ontology. Another approach uses the combination of domain ontologies and SWRL expressions as a starting point for the generation of contract code [53].

Business process execution language (BPEL) [193]: The declarative XML-based Business Process Execution Language (BPEL or WS-BPEL, $n = 1$) is a standard for the specification of executable processes. It relies upon the Web Service Description Standard for service specification and discovery, but adds capabilities for the description of interactions among business processes and web services. Thus, BPEL allows for the description of workflows orchestrating service invocations as seen, for example, in [48], where a BPEL process is translated into a smart contract which calls external web services.

Case management model and notation (CMMN) [242]: CMMN ($n = 1$) is a business process modeling method designed to be complementary to BPMN. As opposed to BPMN, which describes a process as a predefined sequence of activities, CMMN regards a subject in a situation where actions to be taken may need to be chosen and ordered during run-time. CMMN is used, e.g., in [179] for the modeling of common blockchain application patterns, such as oracles and tokenization.

DatalogMTL (DMTL) [43]: Datalog is a logic programming language. Its applications include ontology representation and querying deductive systems. DatalogMTL ($n = 1$) is an extension introducing concepts of metric temporal logic, allowing expression on temporal data. An example for the application of DatalogMTL is the work of Nissl et al. [190], where logic-based smart contracts modeled with DatalogMTL were translated into code executable by Ethereum's virtual machine [272].

Extended entity-relationship model (EER) [104]: The EER model ($n = 1$) includes all concepts of ER, but incorporates several extensions, such as inheritance concepts, clustering of entity types, and complex attributes. EER is used, e.g., in [129] for the representation of an OWL ontology of the block and transaction structure as present in Bitcoin, Ethereum, and Hyperleder Fabric.

Interaction flow modeling language (IFML) [240]: The IFML standard ($n = 1$) offers notations to model interactions of users with front-end applications and the related behavior triggered by user actions or system events. In [100], the authors apply IFML as part of an MDD method for so-called hybrid applications, that involve a DLT system as well as a centralized database. The interactive system component is modeled with IFML. Furthermore, an extension of IFML is used to model workflows.

Object constraint language (OCL) [239] The formal and declarative Object Constraint Language ($n = 1$) allows to specify expressions on UML models. This includes, for example, the definition of queries, invariants, and conditions on operations. The work of Syahputra et al. [235] presents a development method for smart contracts involving REA, UML and OCL. The UML models enriched with OCL expressions serve as input for the generation of code.

Semantic object model (SOM) [90]: The semantic object model ($n = 1$) is an object- and process-oriented modeling method for enterprises. In SOM, a business system is represented on the levels of the enterprise plan, business processes, and resources, especially IT systems. In contrast to ArchiMate or BPMN, SOM describes processes and IT systems in a structural as well as a behavioral view and derives IT systems in their specification by a model-driven method-

ology. Härer [121] applied SOM in conjunction with BPMN for modeling decentralized organizations.

Toronto virtual enterprise ontology (TOVE) [96]: TOVE ($n = 1$) is a formal ontology with a layered structure, containing sub-ontologies for the modeling of various parts of an enterprise, such as activities, products, or organizational structure [97]. Kim et al. [141] adapted TOVE for provenance tracking in supply chain applications. The extended ontology then serves as the foundation for the development of smart contracts.

Unified service description language (USDL) [47]: USDL ($n = 1$) provides means for the description of web services and their relation to underlying business services. Thereby, USDL reverts to an XML-based syntax and includes an extension mechanism. This has been used, e.g., in [27] to adopt USDL for the description of smart contract functionality, aiming to increase the user's trust in contracts for which the implementation is not public.

4.2.2 Employed blockchain technologies

We have identified seven blockchain technologies that have been regarded by academic approaches. In this work, we do not differentiate between the various Hyperleder projects, and instead group them together. An overview on the popularity of the employed blockchain technologies is shown in Fig. 4. By far the most popular technology is Ethereum ($n = 99$), a general purpose blockchain with the capability of executing turing-complete smart contract in a decentralized virtual machine [272]. Numerous approaches did not specify a blockchain technology ($n = 39$). This is exclusively the case for conceptual works that are technology independent. Further, we have found approaches that consider multiple blockchains ($n = 16$). This includes some conceptual works, but more common are multi-chain approaches focusing on the technical level. The second most popular choice is then one of the infrastructure blockchain technologies developed under the umbrella of the Hyperledger project [134] ($n = 15$). These are typically less focused on crypto-economic aspects than general-purpose blockchains, but instead are tailored towards consortial settings, where the cooperation of organizations is enabled through a collectively established blockchain as trust-basis. Hyperledger is followed by another two general purpose blockchain technologies in this ranking, namely Corda [127] ($n = 4$) and Cardano [46] ($n = 2$). Only one approach employs Bitcoin [184]. This low adoption may be explained by the limited smart contract capabilities of the Bitcoin technology.

Since blockchain technologies were not an area of active research prior to 2014, this review covers publications from 2014 onward. The research activity has risen considerably since then, as is evident by the increase of publications over the years, shown in Fig. 5.

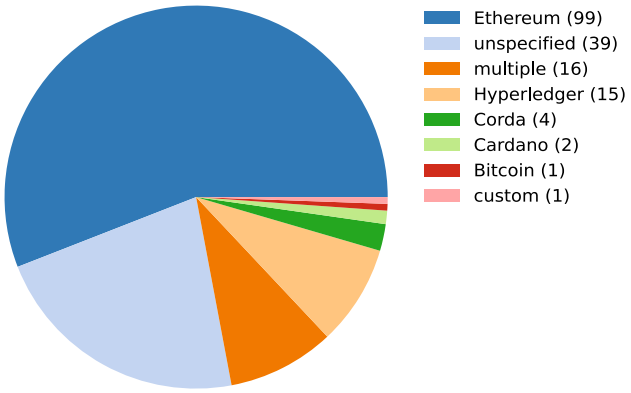

Fig. 4 Employed blockchain technologies of the approaches in the final corpus. Ethereum is by far the most popular choice in this set. However, numerous conceptual approaches do not rely on a specific blockchain technology

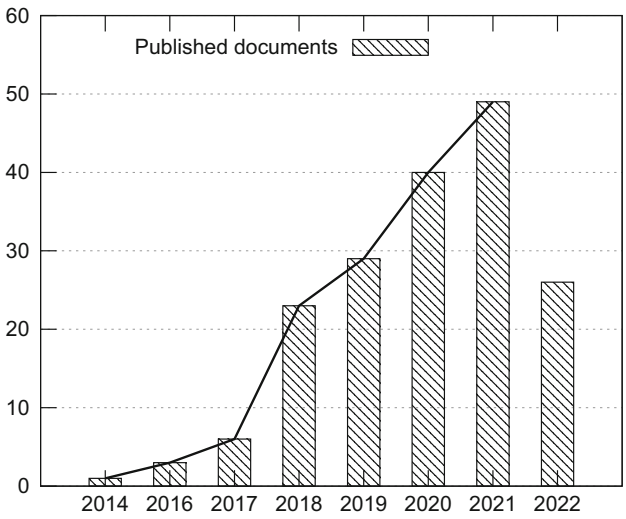

Fig. 5 Number of documents in the final corpus grouped by year. This includes preprints but no surveys. Note that the review does not cover the entirety of 2022

4.2.3 Nature of realization

All documents in the final corpus have been grouped by their realization, that is, in *conceptual* and *executable code generating*.

Conceptual approaches: Documents presenting conceptual approaches are shown in Table 3. Most commonly, these conceptual approaches do not relate to a specific blockchain technology ($n = 39$). This is not surprising, as conceptual works generally allow for a higher degree of abstraction. However, some approaches are technology-specific. In these cases, Ethereum is the most popular blockchain technology ($n = 17$), followed by Hyperledger ($n = 5$), Corda ($n = 1$), and Bitcoin ($n = 1$). UML is the most prominently employed modeling language in this group ($n = 25$), e.g., as part of a methodology or for the design of metamodels.

Table 3 Publications proposing a conceptual method for the design of blockchain applications, or reasoning thereof. That is, the method does not automatically generate code artifacts executable in the context of a specific blockchain technology. Instead, such approaches may focus on the conceptualization of the blockchain domain, resulting in a model representation, reasoning on blockchain systems based on models, conceptual language mappings, or model-based methodologies for the design and development of blockchain applications. Conceptual artifacts include but are not limited to ontologies, metamodels, methodologies, semantic models, business models, and model-based verification and simulation without code generation

Base language	BT	References	Base language	BT	References
AOM	u	LiBin et al. [153]	dsl, OWL, UML	ETH	Dwivedi et al. [80]
AOM	u	Udokwu et al. [253]	e^3 value	u	Perrelet et al. [198]
AOM, UML	u	Udokwu et al. [255]	e^3 value	u	Poels et al. [203]
ArchiMate	m	Curty et al. [66]	EER, OWL	m	Hector et al. [129]‡
ArchiMate	u	Jiang et al. [135]	i*	u	Hamadi et al. [118]
ArchiMate, BPMN, UML	m	Ellervee et al. [82]	i*, UML	u	Vingerhoets et al. [258]
BPMN	HL	Panduwinata et al. [196]	i*, UML	u	Vingerhouts et al. [259]
BPMN, CMMN	u	Milani et al. [179]	OWL	ETH	Bella et al. [26]
BPMN, DEMO	HL	Guerreiro et al. [114]	OWL	u	Scrocca et al. [219]‡
BPMN, DMN	ETH	Nousias et al. [192]	OWL, RDF, UML	u	Amato et al. [10]
BPMN, dsl, Petri Net	HL	Dittmann et al. [77]	OWL, SWRL	ETH	Besançon et al. [29]
BPMN, e^3 value	ETH	Gómez et al. [105]	OWL, SWRL	u	Choudhury et al. [53]
BPMN, ER, UML	u	Rocha et al. [208]	Petri Net	ETH	He [126]
BPMN, OWL	u	Besançon et al. [28]	Petri Net, UML	u	Dwivedi et al. [78]
DEMO	ETH	Aparício et al. [13]	Petri Net, UML	u	Kherbouche et al. [140]
DEMO	ETH	Aparício et al. [14]	Petri Net, UML	u	Ladleif et al. [144]
DEMO, REA, UML	u	de Kruijff et al. [69]	RDF	ETH	Petrović et al. [201]
DEMO, REA, UML	u	de Kruijff et al. [70]	RDF	u	Seebacher et al. [220]
DEMO, UML	HL	Silva et al. [223]	RuleML	u	de Kruijff et al. [71]
dsl	BC	Andrychowicz et al. [11]	RuleML	u	de Kruijff et al. [72]
dsl	HL	Bollen [37]	TOVE	ETH	Kim et al. [141]
dsl	m	Six et al. [225]	UML	CO	Górski et al. [110]
dsl	u	Amaral de Sousa et al. [8]	UML	ETH	Cano-Benito et al. [44]
dsl	u	Amaral de Sousa et al. [9]	UML	ETH	Lallai et al. [147]
dsl	u	Bai et al. [22]	UML	ETH	Marchesi et al. [168]
dsl	u	Barisic et al. [24]	UML	ETH	Marchesi et al. [170]
dsl	u	Conchon et al. [58]	UML	ETH	Olivé [194]
dsl	u	He et al. [125]	UML	ETH	Teruel et al. [237]
dsl	u	Park et al. [197]	UML	u	Górski [109]
dsl	u	Purnell et al. [204]	UML	u	Jurgelaitis et al. [138]
dsl	u	Regnath et al. [207]	UML	u	Roussille et al. [211]
dsl	u	Shi et al. [222]	UML	u	Sánchez-Gómez et al. [214]
dsl	u	Tsai et al. [251]	UML, USDL	ETH	Ben Slama Souei et al. [27]

AOM agent-oriented model, *BPMN* business process model and notation, *CMMN* case management model and notation, *DEMO* design and engineering methodology for organizations, *DMN* decision model and notation, *ER* entity-relationship, *EER* extended ER, *OWL* web ontology language, *RDF* resource description framework, *TOVE* Toronto virtual enterprise ontology, *UML* unified modeling language, *USDL* unified service description language, *dsl* domain-specific language, *BT* blockchain technology, *ETH* Ethereum, *CA* Cardano, *CO* Corda, *HL* Hyperledger, *c* custom, *m* multiple, *u* unspecified

‡Preprint or technical report

Table 4 Publications proposing an executable code generating method. That is, the method automatically or partially automatically generates code artifacts from some model representation, and the artifacts are executable in the context of a specific blockchain technology. Such artifacts include but are not limited to smart contracts, blockchain connectors, integration facilities, UI components for blockchain applications, and process or workflow encodings

Base language	BT	References		Base language	BT	References	
ArchiMate, DEMO	HL	Babkin et al.	[19]	dsl	ETH	Bistarelli et al.	[33]
BPEL	ETH	Carminati et al.	[48]	dsl	ETH	Boubeta-Puig et al.	[39]
BPMN	ETH	Abid et al.	[2]	dsl	ETH	Boychenko et al.	[40]
BPMN	ETH	Azzopardi et al.	[18]	dsl	ETH	Chen et al.	[51]
BPMN	ETH	Bagozi et al.	[20]	dsl	ETH	Dharanikota et al.	[73]
BPMN	ETH	Brahem et al.	[41]	dsl	ETH	Dwivedi et al.	[79]
BPMN	ETH	Corneli et al.	[59]	dsl	ETH	Frantz et al.	[98]
BPMN	ETH	Corneli et al.	[60]	dsl	ETH	Franz et al.	[99]
BPMN	ETH	Corradini et al.	[61]	dsl	ETH	Henry et al.	[130]
BPMN	ETH	Corradini et al.	[62]	dsl	ETH	Kolb et al.	[143]
BPMN	ETH	Corradini et al.	[64]	dsl	ETH	Liu, C. et al.	[154]
BPMN	ETH	Corradini et al.	[65]	dsl	ETH	Liu, C. et al.	[155]
BPMN	ETH	Di Ciccio et al.	[55]	dsl	ETH	Liu, C. et al.	[156]
BPMN	ETH	Klinger et al.	[142]	dsl	ETH	Liu, Y. et al.	[158]
BPMN	ETH	Ladleif et al.	[146]	dsl	ETH	Madsen et al.	[166]
BPMN	ETH	López-Pintado et al.	[159]	dsl	ETH	Mao et al.	[167]
BPMN	ETH	López-Pintado et al.	[160]	dsl	ETH	Marchesi et al.	[169]
BPMN	ETH	López-Pintado et al.	[162]	dsl	ETH	Mavridou et al.	[172]
BPMN	ETH	Loukil et al.	[164]	dsl	ETH	Mavridou et al.	[173]
BPMN	ETH	Morales-Sandoval et al.	[181]	dsl	ETH	Mavridou et al.	[174]
BPMN	ETH	Schindelmann et al.	[217]	dsl	ETH	Meng et al.	[177]
BPMN	ETH	Spalazzi et al.	[229]	dsl	ETH	Nelaturu et al.	[186]
BPMN	ETH	Sturm et al.	[232]	dsl	ETH	Nelaturu et al.	[187]
BPMN	ETH	Sturm et al.	[233]	dsl	ETH	Rosa-Bilbao et al.	[210]
BPMN	ETH	Tonga Naha et al.	[247]	dsl	ETH	Sergey et al.	[221]‡
BPMN	ETH	Tran et al.	[249]	dsl	ETH	Suvorov et al.	[234]‡
BPMN	ETH	Weber et al.	[263]	dsl	ETH	Tan et al.	[236]
BPMN	HL	Alves et al.	[7]	dsl	ETH	Trebbau et al.	[250]
BPMN	m	Bagozi et al.	[21]	dsl	ETH	Weingärtner et al.	[265]
BPMN	m	Corradini et al.	[63]	dsl	ETH	Wickramarachchi et al.	[267]
BPMN	m	Falazi et al.	[87]	dsl	ETH	Wöhrer et al.	[270]
BPMN	m	Falazi et al.	[88]	dsl	ETH	Wöhrer et al.	[271]
BPMN	m	Ladleif et al.	[145]	dsl	HL	Astigarraga et al.	[16]
BPMN, DEMO, dsl	ETH	Skotnica et al.	[227]	dsl	HL	Bore et al.	[38]
BPMN, DEMO, dsl	m	Skotnica et al.	[226]	dsl	HL	Merlec et al.	[178]
BPMN, DMN	ETH	Haarmann et al.	[116]	dsl	HL	Mirković et al.	[180]
BPMN, dsl, Petri Net	ETH	López-Pintado et al.	[161]	dsl	m	Hamdaqa et al.	[119]
BPMN, dsl, Petri Net	ETH	López-Pintado et al.	[163]	dsl	m	Hamdaqa et al.	[120]
BPMN, dsl, UML	ETH	Skotnica et al.	[228]	dsl	m	Qasse et al.	[205]‡
BPMN, Petri Net	ETH	García-Bañuelos et al.	[102]	dsl	m	Qasse et al.	[206]
BPMN, SCXML	HL	Nakamura et al.	[185]	dsl	u	Abbas et al.	[1]
BPMN, SOM	ETH	Härer	[121]	dsl, RuleML	ETH	van den Heuvel et al.	[256]
BPMN, UML	ETH	Lu et al.	[165]	dsl, UML	m	Heckel et al.	[128]
BPMN, UML	ETH	Sturm et al.	[231]	dsl, UML, SCXML	HL	Sato et al.	[216]

Table 4 continued

Base language	BT	References		Base language	BT	References	
DatalogMTL	ETH	Nissl et al.	[190]	ER, IFML, UML	HL	Fraternali et al.	[100]
DMN	ETH	Haarmann	[115]	i*	ETH	Tsiounis et al.	[252]
DMN	ETH	Haarmann et al.	[117]	OCL, REA, UML	m	Syahputra et al.	[235]
dsl	CA	Lamela Seijas et al.	[148]	Petri Net	c	Evermann et al.	[84]
dsl	CA	Lamela Seijas et al.	[149]	Petri Net	ETH	Zupan et al.	[277]
dsl	ETH	Allouche et al.	[5]	UML	CO	Górski et al.	[111]
dsl	ETH	Asawa et al.	[15]	UML	CO	Górski et al.	[112]
dsl	ETH	Azzopardi et al.	[17]	UML	CO	Górski et al.	[113]
dsl	ETH	Baresi et al.	[23]‡	UML	ETH	Garamvölgyi et al.	[101]
dsl	ETH	Biryukov et al.	[30]	UML	ETH	Jurgelaitis et al.	[137]
dsl	ETH	Bistarelli et al.	[31]	UML	HL	Jurgelaitis et al.	[139]
dsl	ETH	Bistarelli et al.	[32]				

BPEL business process execution language, *BPMN* business process model and notation, *DEMO* design and engineering methodology for organizations, *DMN* decision model and notation, *ER* entity-relationship, *IFML* interaction flow modeling language, *OCL* object constraint language, *REA* resource-event-agent ontology, *SCXML* state chart extensible markup language, *SOM* semantic object model, *UML* unified modeling language, *dsl* domain-specific language, *BT* blockchain technology, *ETH* Ethereum, *CA* Cardano, *CO* Corda, *HL* Hyperledger, *c* custom, *m* multiple, *u* unspecified
‡Preprint or technical report

Some approaches propose or revert to domain-specific languages ($n = 16$). This includes, for example, declarative legal languages, notations for state machines or automata, and non-standard specification languages. Semantic web languages, including OWL, RDF, SWRL, RuleM, and USDL ($n = 13$) are used for example to author domain ontologies.

Executable code generating approaches: Documents presenting approaches with the capability of generating executable code are shown in Table 4. The most used blockchain technology in this group is by far Ethereum ($n = 82$), followed by Hyperledger ($n = 10$), Corda ($n = 3$), and Cardano ($n = 2$). Twelve ($n = 12$) approaches support or target multiple blockchains. This is achieved either by directly implementing the necessary code translation and connectors or by targeting an intermediary language that is compatible with multiple blockchain technologies, for example, the DAML language (c.f. Table 6) as used in [119]. Most code generating approaches apply domain-specific languages for modeling ($n = 58$), ranging from declarative textual to visual diagrammatic representations. In this group, BPMN is the most commonly used standard modeling language ($n = 42$), followed by UML ($n = 13$).

4.3 Content-based computational analysis

In addition to the manual literature review we conducted a content-based computational analysis of the derived set of relevant papers. For this purpose we reverted to Latent Dirichlet Allocation (LDA), an established topic modeling approach for text summarization in the field of information retrieval [50]. We follow the process of a computational analysis based on LDA as established and successfully applied before in [123, 182, 195].

Topic modeling techniques have been developed since the 1990s stemming from Latent Semantic Indexing and similar approaches, leading to research on LDA in the 2000s [34, 35]. LDA introduced the identification of topics from a document collection by training a model of topic-word distributions, resulting in the generation of k topics consisting of word occurrences. Today, LDA with optimizations such as Gibbs sampling is considered an online topic modeling with high performance characteristics [54], in addition to graph-based, biterm, and self-aggregating methods as well as further runtime optimizations such as GPU acceleration [50, 54].

In particular, LDA allows for the generation of k topics for a document collection such that each document is characterized by its own topic distribution θ_d, representing multiple topics per document in the underlying model. The model of topic-word distributions is trained through expectation maximization, maximizing the likelihood that the topic distributions of each document $\theta_{d,k}$ form an overall topic-word distribution of k topics in the collection [34]. Furthermore, the model is characterized by the ratio of topics per document (α) and the ratio of words per topic (β). As a result, each of the k topics is represented by the top n words according to the weight, corresponding to occurrences in the probabilistic model. In terms of software, the LDA implementation of MALLET (MAchine Learning for LanguagE Toolkit[1]) was used in RapidMiner 9.5 with optimizations for Gibbs sampling [188] and concurrent processing [273].

[1] http://mallet.cs.umass.edu/topics.php.

Topic 1	
Word	**Weight**
process	3053
business	2134
chain	1218
technology	1188
activity	783
Sum	**8376**

Topic 2	
Word	**Weight**
process	3065
execution	1141
event	1062
bpmn	678
task	601
Sum	**6547**

Topic 3	
Word	**Weight**
transition	1316
state	1292
function	723
property	634
verification	633
Sum	**4598**

Topic 4	
Word	**Weight**
transaction	2042
workflow	718
participant	466
state	405
system	391
Sum	**4022**

Topic 5	
Word	**Weight**
language	858
rule	646
party	555
legal	551
code	540
Sum	**3150**

Topic 6	
Word	**Weight**
application	742
code	645
ethereum	573
user	557
function	554
Sum	**3071**

Topic 7	
Word	**Weight**
supply	739
asset	733
chain	557
product	507
food	349
Sum	**2885**

Topic 8	
Word	**Weight**
choreography	1239
message	405
execution	345
participant	327
chorchain	253
Sum	**2569**

Topic 9	
Word	**Weight**
ontology	696
element	469
agent	386
requirement	336
diagram	333
Sum	**2220**

Topic 10	
Word	**Weight**
uml	484
code	433
class	379
deployment	372
transformation	329
Sum	**1997**

Fig. 6 Final results of the computational analysis using LDA with a set of 10 topics based on five terms and the final weights assigned by the algorithm. The final set of topics displayed here was then subjected to a manual refinement (see Sect. 4.4) over three iterations. These iterations consisted of the identification of topics, their assignment to papers by three raters, the calculation of inter-rater reliability measurements, joint discussions of the results involving manual screening of the papers in unclear cases

We apply LDA as part of a data mining process, consisting of *data selection*, *preprocessing*, *transformation*, and *data mining* as suggested initially by the KDD model [89]. Data selection concluded with the full text PDF documents resulting from the literature review, including a small set of survey papers (c.f. Table 1) and few relevant papers only published as report or on ArXiv (c.f. Tables 3 and 4 marked with ‡) for generating an initial set of topics computationally through LDA. With this collection, preprocessing and transformation of the PDF documents to text documents followed. In preprocessing, data cleaning and related operations took place, in particular the tokenization of documents, minimal stemming, synonym replacement, cleaning for the removal of stop words, incomplete data, and irrelevant article fragments such as header texts or comments.

The LDA was carried out over multiple iterations. In particular, for determining the initial k topics, iterations for $k \in [6, 14]$ were calculated first, followed by defining domain-related stop words and synonyms. The results were evaluated manually by the authors with the criterion being to avoid overly generic or specific topics and words, concluding with the selection of $k = 10$ topics. The topics that resulted from this initial application of LDA are shown in Fig. 6.

4.4 Manual topic refinement and synthesis by rating and qualitative assessment

In a next step we refined the topics proposed by LDA, screened and assigned papers to the resulting topics manually over multiple iterations with inter-rater reliability measurements, leading to a synthesis of computationally and manually determined topics.

The initial step consisted of identifying suitable labels for each topic in a joint discussion by the authors and the combination of similar topics. Given the topics, each subsequent iteration consisted of manually assigning papers to topics by three raters, the authors of this paper, the calculation of inter-rater reliability measurements, joint discussions of the results, and screenings of paper abstracts in unclear cases. The following inter-rater reliability indicator was measured as a qualitative assessment concerning the percentage agreement and Cohen's Kappa [57, 176] over the assignments of $n = 177$ papers in the collection. (1) The unanimous agreement $A_U = {}^1/_n * \sum_{i=1}^{n} u_i$ where $u_i \in [0, 1]$ is the agreement for the assignment of paper i with $u_i = 1$ if all raters made the same assignment and $u_i = 0$ if assignments differed. (2) The mean agreement $A_\mu = {}^1/_n * \sum_{i=1}^{n} {}^{v_i}/_3$ where $v_i \in [0, 3]$ is the number of raters in agreement for the topic assignment of paper i. (3) Cohen's Kappa κ was calculated according to the procedure described by Cohen and McHugh [57, 176], accounting for agreements due to chance.

In the first of three iterations, the initial assignment resulted in $A_U = 62.1\%$ of the papers assigned to the same categories by all raters with a mean agreement $A_\mu = 72.5\%$, indicating moderate to high agreement below the threshold of $A_\mu = 75\%$ that might be considered sufficient [191]. When accounting for chance with Cohen's Kappa, $\kappa = 0.60$ was reached, considered slightly below a *substantial agreement* at $\kappa = 0.61$ or above, according to Cohen and Landis et al. [57, 150]. In the joint discussion, papers not assigned to categories by at least two reviewers were identified first for discussing the scope and suitability of topics. This concerned 8 papers that were determined out of scope for the existing topics and assigned to the newly added topics *Reference Models* and *Business Modeling*. Secondly, under-represented or unclear

Table 5 Final set of topics resulting from computational analysis (Sect. 4.3) and three iterations of manual refinement (Sect. 4.4). For each topic, the number of papers that were assigned to the topic with unanimous agreement by all raters is given. The inter-rater reliability measurements of unanimous agreement $A_U = 72.3\%$, mean agreement $A_\mu = 79.3\%$, and Kappa $\kappa = 0.71$ indicated very high agreement (see Sect. 4.4)

Topic	Number of assigned papers
Process, workflow, choreography, and decision models	48
Application development	42
Formal aspects and verification	15
Legal aspects, rules, and languages	9
Ontology	7
UML modeling	5
Business modeling	4
Reference models	3
Supply chain	2

aspects for existing topics were discussed when raters had noted additional comments during the rating. This discussion revealed unclear assignments for papers on decision modeling, which were considered part of a process-centered topic *Process, Workflow, Choreography, Decision Models* as result of the discussion. Furthermore, formal verification papers were grouped with papers on further formal aspects, such as formal specification, in a topic *Formal Aspects and Verification*.

In the second iteration, unanimous agreement $A_U = 72.3\%$, mean agreement $A_\mu = 79.3\%$, and Kappa $\kappa = 0.71$ were reached, indicating relatively high agreement on the assignment of topics and *substantial agreement* regarding κ. The joint discussion did not reveal topics unclear in scope or suitability, with all papers being assigned by at least two raters. Based on the rater's comments, unclear aspects during the assignment led to marking related papers for an additional abstract screening as input for the last iteration.

In the third iteration, after screening abstracts in 23 unclear cases, the final measurements indicate unanimous agreement $A_U = 76.3\%$, mean agreement $A_\mu = 83.1\%$, and Kappa $\kappa = 0.75$. At this point, agreement on topics among the raters was considered very high, even when accounting for agreement by chance through κ.

Table 5 shows the final set of topics with the number of papers assigned through unanimous agreement. The details of these topics will be discussed in Sect. 6 by highlighting the core papers found for each topic.

5 Industrial low-code and no-code software platforms

In the following, we continue in our research methodology by reviewing state-of-the-art low-code and no-code software platforms towards their suitability for blockchain application development. The review process is described in Fig. 7.

Low-code and no-code software platforms are increasingly available for practitioners to develop applications using abstractions beyond source code [36, 76]. While low-code

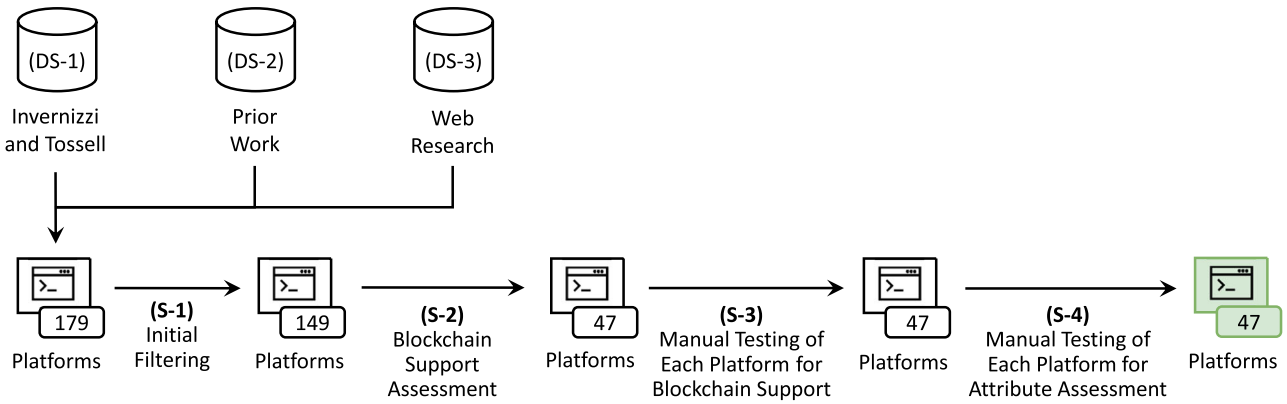

Fig. 7 Systematic review process for the identification of industrial low-code and no-code software platforms supporting blockchain application development. The data sources consist of (DS-1), an existing compilation by Invernizzi and Tossell (see Sect. 5.1), (DS-2), prior work [122], and (DS-3), additional research on the web. After (S-1), an initial filter step leaving reachable and sufficiently described platforms, (S-2) assessed blockchain support, concluding in (S-3) and (S-4), where platforms were tested with small example implementations for classifying in (S-3) their blockchain capabilities and for determining application attributes (S-4) such as open source availability

platforms are in their representation still close to the source code and often suited for technical users and developers, no-code platforms go one step further by providing representations on an abstraction level above source code that are accessible to non-technical domain experts and even *citizen developers* [209]. For example, low-code approaches might provide an interactive drag-and-drop environment for arranging blocks of statements in a procedural manner, while no-code approaches often guide users through dialogs and allow arranging pre-configured user interface (UI) components. Since many platforms are today web-based, their ease of use also stems from *integrations*, by connecting to existing cloud-services or websites and integrating their functionality [213].

5.1 Review process

Relying on expert knowledge, the review considers as a starting point an informal compilation by Invernizzi and Tossell[2] that identified 145 web-based platforms such as website and app builders, e-commerce services, and data dashboards. The identified solutions differ substantially in the scope and applications they target. We review solutions of the compilation, extended with approaches from prior research and additional investigations of state-of-the-art platforms in October 2022. Therefore, the data sources for this review are (DS-1): the compilation by Invernizzi and Tossell, (DS-2): practical approaches from prior research [122], and (DS-3): additional research on blockchain-specific low-code and no-code solutions available on the web.

We applied a four-step process, consisting of an initial filtering step (S-1), the evaluation of scope and applicability for blockchains in step (S-2), the classification of solutions applicable to blockchains (S-3), and the characterization through representation and application attributes (S-4). Initially, 179 solutions were identified. In (S-1), we manually retrieved descriptions from the vendor websites in addition to information provided by (DS-1), followed by filtering out duplicate entries, those that could not be reached on the web, or did not provide sufficient information on their websites (e.g. closed beta software). The remaining 150 solutions were evaluated in (S-2) regarding their scope of blockchain integration. Finally, 47 solutions were identified as applicable for blockchains. For these platforms, further evaluations for assigning (S-3) categories and (S-4) the type of the predominant representation and application attributes were carried out.

[2] https://pinver.medium.com/decoding-the-no-code-low-code-startup-universe-and-its-players-4b5e0221d58b.

5.2 Results

For discussing available platforms and their blockchain integration, we distinguish between *1st degree* and *2nd degree* integration. A platform supports 1st degree integration if it interacts directly with blockchains through its software or services. 2nd degree integration is supported if an external service could be integrated that offers 1st degree integration. The criteria for the selected platforms (S-3) listed in Table 6 are that they (a) offer blockchain integration of 1st or 2nd degree and (b) were considered a low-code or no-code approach. For (S-4), the table outlines the 9 categories identified in this step along with a description of application attributes, 1st and 2nd degree blockchain integration (columns d_1 and d_2), open source (column s) availability of the implementation, and the representation (column r) primarily used to abstract from source code.

5.2.1 Platforms with 1st degree integration

1st degree blockchain integration has been found in 18 solutions intended for building mobile apps, web apps and websites, workflow integration and automation, and smart contract development.

App builders: The primary integration features in the categories for app builders, concerning platforms primarily for mobile applications (AM) or web applications (AW), are the creation of decentralized apps (DApps) and the integration of cryptocurrency-related data, e.g., price information. Platforms such as Outsystems (AW-6) and Bubble (AW-1) support DApps, where components of a mobile, desktop, or web app can send blockchain transactions and call smart contract functions, e.g., through web3 plugins such as the MetaMask browser extension.[3] In the case of Outsystems, web3 functionality through MetaMask is available in pre-defined components provided as part of the platform's integrated development environment (IDE). Following the web3 concept [45], apps or websites are created using decentralized platforms and infrastructures that depend less on large, centralized, and potentially influential entities such as cloud providers which could represent potential single points of failure. This might be achieved by blockchain transactions made through MetaMask or by decentralized data retrieved from Ethereum.

An example for the development of a blockchain-based app for supply chain tracking is shown in Fig. 8 as demonstrated before by the authors in [67]. In this category, representations abstract from source code (column r) and often use flow-based editors. Elements connected by flow relationships specify, e.g., the navigation through user interfaces, computational steps, or other actions carried out

[3] https://metamask.io/.

Table 6 Industrial low- and no-code approaches with 1st or 2nd degree blockchain integration

Cat-ID	Name	Website	Description	d_1	d_2	s	r
AM-1	Adalo	adalo.com	Apps and websites, creation of spreadsheets	–	+	○	u
AM-2	Axonator	axanator.com	Apps, additional workflows and dashboards	–	+	–	f
AM-3	BuildFire	buildfire.com	Apps with e-commerce and industry-specific focus	–	+	○	u
AM-4	Glide	gildeapps.com	Apps, esp. data-centric apps with tabular data, BC temp	+	+	–	u
AW-1	Bubble	bubble.io	Websites and cloud apps, small business focus, web3 plugins	+	+	–	u
AW-2	Builder.ai	builder.ai	Websites and progressive web apps, BC temp	+	–	–	u
AW-3	Draftbit	draftbit.com	Websites and apps, mobile use focus, industry-specific temp	–	+	○	u
AW-4	Landbot	landbot.io	Chatbot builder, specification using visual flows	–	+	–	u
AW-5	Mendix	mendix.com	Complex apps, websites, cloud services, multi-view UI, IDE	–	+	–	f
AW-6	Outsystems	outsystems.com	Complex apps, websites, cloud services, multi-view UI, IDE, web3	+	–	–	f
D-1	Levity	levity.ai	Predictive analytics for documents, images, text	–	+	–	f
D-2	Obviously AI	obviously.ai	Predictive analytics for tabular data and databases	–	+	–	d
D-3	Parabola	parabola.io	Analytics for data flows from multiple sources, sales data focus	–	+	–	f
F-1	Arengu	arengu.com	Forms and dialog flows, components for e.g. authentication, payments	–	+	○	d
F-2	Formstack	formstack.com	Forms using dialogs with if-then-logic	–	+	–	d
F-3	Tally	tally.so	Forms using documents and integration	–	+	–	d
IN-1	Budibase	budibase.com	Tools esp. for tabular data of ERP, HR, sales, customers	–	+	–	u
IN-2	Jet Admin	jetadmin.io	Tools esp. for administration, CRM, approval workflows	–	+	○	u
IN-3	Windward	windwardstudios.com	Tools for building reports and dashboards, document generation	–	+	–	d
SC-1	DAML	daml.com	SC generation, DSL for contract logic, tokens, assets	+	–	+	t
SC-2	Dappbuilder	dappbuilder.io	Small decentralized apps with websites and SCs, pre-defined temp	+	–	+	d
SC-3	Simba Chain	simbachain.com	SC generation, graph-based data models, generation of data structures	+	–	–	t
SP-1	Actiondesk	actiondesk.io	Spreadsheet-based apps, focus on integration with apps, reports	–	+	–	s
SP-2	Airtable	airtable.com	Spreadsheet-based apps, multiple views, temp., data integrations	–	+	–	s

Table 6 continued

Cat-ID	Name	Website	Description	d_1	d_2	s	r
SP-3	Rows	rows.com	Spreadsheet-based apps, esp. database, web API access, event triggers	−	+	−	s
WA-1	Aurachain	aurachain.ch	Process automation with database and API access, BC temp	+	−	−	d
WA-2	AWS	aws.amazon.com	Workflow Studio, flows for cloud services, lambda functions, serverless	+	+	−	f
WA-3	Azure	azure.microsoft.com	Azure Logic, visual flows for services, lambda functions, user-facing apps	−	+	−	f
WA-4	IFTTT	ifttt.com	If-then workflows, esp. web apps and services, BC services	+	+	−	d
WA-5	Make	www.make.com	Integration, esp. of existing services, event-based, BC services	+	+	−	d
WA-6	Kissflow	kissflow.com	Integration using temp. components, esp. reports, cases, apps	−	+	−	f
WA-7	n8n	n8n.io	Integration, esp. w1eb apps and services, data processing, BC services	+	−	+	f
WA-8	NodeRed	nodered.org	Integration using components for logic, esp. for cloud services, IoT, BC	+	+	+	f
WA-9	Pipefy	pipefy.com	Integration, esp. pre-defined ERP functions such as accounts and HR	−	+	−	f
WA-10	Process Str	process.st	Business processes and workflows, document-based workflows	−	+	−	f
WA-11	Tray	tray.io	Integration, esp. web services, enterprise and e-commerce focus	−	+	−	f
WA-12	Waylay	waylay.io	Integration, esp. IoT, digital twins, process automation, data processing	−	+	−	f
WA-13	Workato	workato.com	Integration, esp. of business cloud services	−	+	−	f
WA-14	Zapier	zapier.com	Integration, esp. of existing web apps and services, BC services	+	+	○	d
WB-1	Atra	atra.io	Websites and web3 apps, UI focus, plugins for cryptocurrency wallets	+	−	○	f
WB-2	ICME	icme.io	Websites, temp.-based, builder and websites hosted on Dfinity BC	+	−	−	u
WB-3	Pory	pory.io	Websites and small apps, integrations from external sites and services	−	+	−	f
WB-4	Softr	softr.io	Websites and small apps, integrations from external sites and services	−	+	−	f
WB-5	Squarespace	squarespace.com	Websites, UI focus, dialogs and visual flows for specification, temp	−	+	−	u
WB-6	Unstack	unstack.com	Websites, UI focus, focus on landing pages and e-commerce	−	+	−	u
WB-7	Webflow	webflow.com	Websites with complex structures, BC temp. for e-commerce	+	+	−	u
WB-8	Xooa	xooa.com	Websites and web3 apps, pre-defined components, dashboards	+	−	○	d

A platform denoted to have 1st degree integration interacts directly with supported blockchains through its software or services. 2nd degree integration refers to the support of an integration with an external service that offers 1st degree integration
Cat: category (AM: app builder with mobile focus, AW: app builder with web focus, D: data, F: form builder, IN: internal tools, SC: smart contract development, SP: spreadsheet tools, WA: workflow integration and automation tools, WB: website builder), d_1: 1st degree blockchain integration, d_2: 2nd degree blockchain integration, s: open source implementation, r: primary representation abstract from code (u: UI-based, f: flow-based, d: dialog-guided, s: spreadsheet-based, t: text-based)
+: applicable, ○: partially applicable, −: not applicable, app: application, temp: template

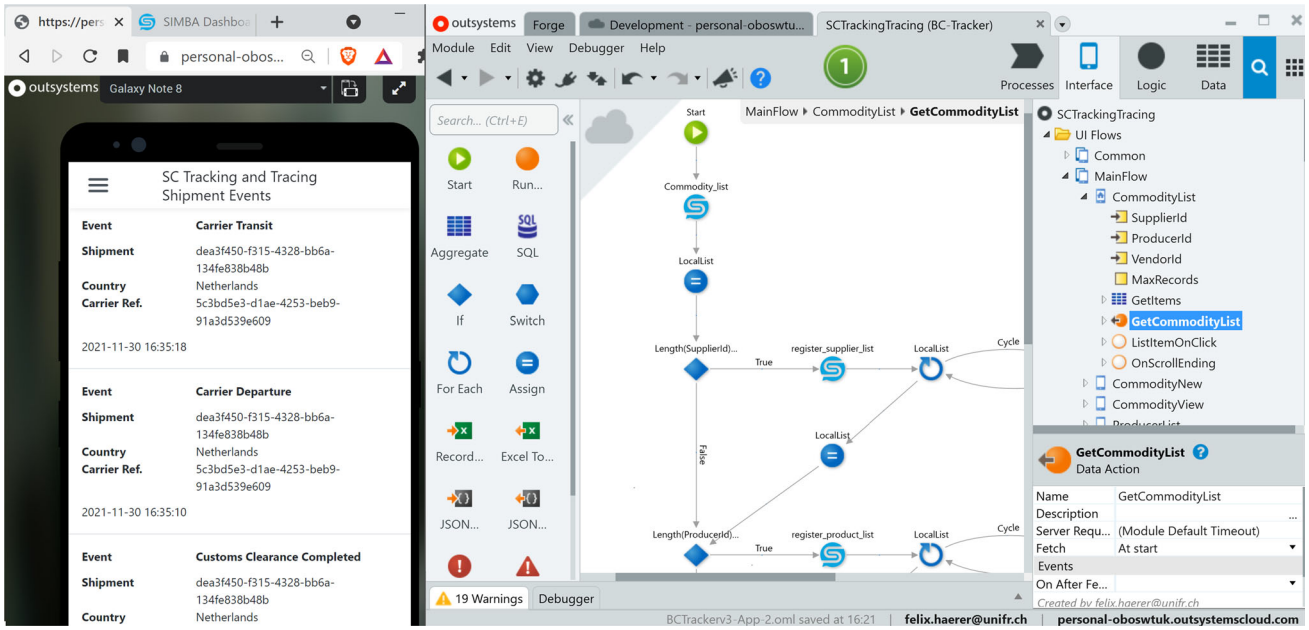

Fig. 8 Outsystems studio, a low-code application builder for web and mobile applications showing an integrated development environment using visual flows (right-hand side), here specifying the user interactions and processing steps in a supply chain tracking application, along with an android emulator executing the app (left-hand side) with retrieval of blockchain data

sequentially or non-sequentially using conditional branches or events. In the example, a list of goods or so-called *commodities* is retrieved from the Ethereum blockchain, stored in a local list, and—for each supplier—processed regarding shipment and product data.

Website builders: For website builders, blockchain integration has only been found for integrating cryptocurrency-related data, with the exception of ICME (WB-2). ICME is a website builder for creating websites on the Dfinity blockchain. The app and the resulting websites are hosted on Dfinity, such that the creation of the website as well as the hosting do not rely on traditional client–server-architectures and instead retrieve and process data on nodes of the decentralized blockchain network.

Workflow integration and automation tools: Tools in this area automate and execute user-defined workflows and allow for integration. This concerns on the one hand cloud providers such as Amazon Web Services (AWS) (WA-2) integrating services for cloud computing. On the other hand, there exist platforms focused on the integration of web apps and web services. For instance, n8n (WA-7), and Zapier (WA-14)[4] realize this approach.

AWS (WA-2) and Microsoft Azure (WA-3) are cloud providers offering low-code solutions in combination with their cloud services, where AWS offers 1st degree integration in supporting service calls to the Amazon Quantum Ledger Database (QLDB) and Amazon Managed Blockchain. For AWS and Azure, flow-based representations are used for specifying calls to services and lambda functions. Similar to the flows shown in Fig. 8, they visualize the subsequent invocation of steps for execution along with execution logic. Using cloud services, steps are specified, possibly depending on conditions or events, for invoking programs of platform-as-a-service offerings with corresponding computation and data storage steps of infrastructure-as-a-service components. Lambda functions allow for specifying functions abstract from concrete servers or services offering scalability and distribution advantages not tied to the implemented source code or the servers. AWS relies on a state machine representation called *step functions* for this purpose that follows the serverless compute paradigm.

Azure allows for lambda functions through Azure Logic Apps that describe application logic in a flow-based manner. Furthermore, Azure allows the connection and integration with user-facing apps, branded power apps, developed in a no-code environment through the composition of UI elements.

Apart from cloud providers, platforms focused on the integration of web apps and web services realize applications on the principle of so-called integrations, by connecting existing apps and services in an automated workflow. In this area, simple workflows might be specified with platforms such as If-This-Then-That (IFTTT) (WA-4) that conditionally call services based on the output of other platforms, e.g. reading

[4] In addition to platforms supporting 2nd degree integration WA-4 to WA-14.

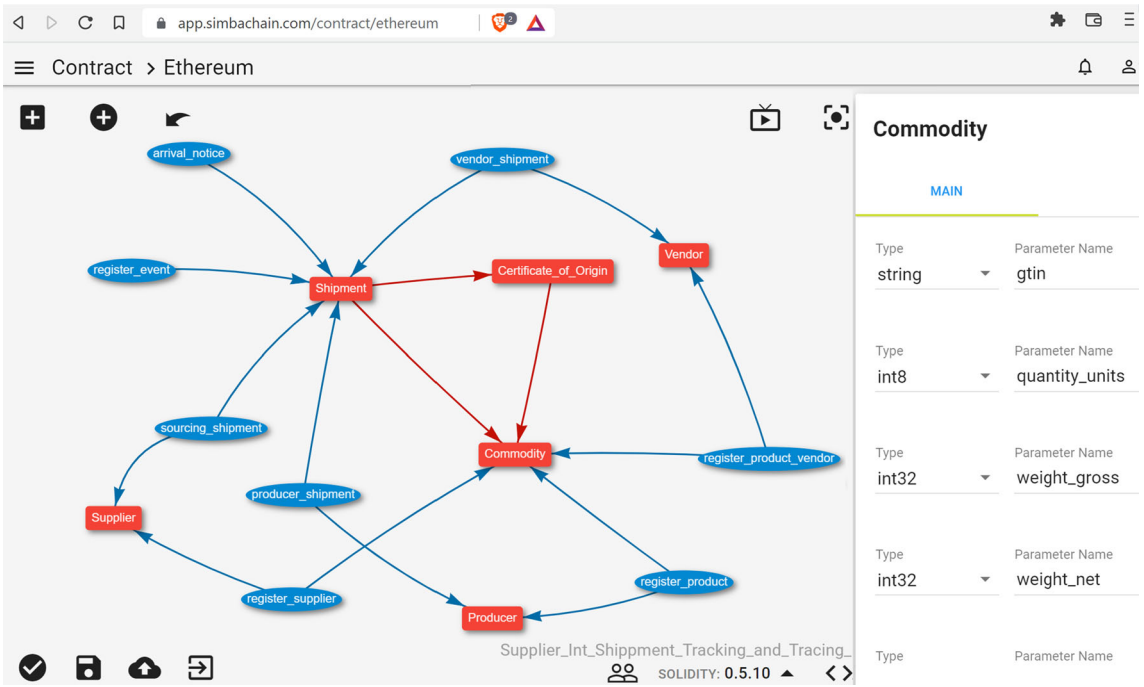

Fig. 9 Simba Chain, a no-code smart contract development platform showing a data model consisting of assets (rectangles) and transactions (ellipses) that result in the generation of a smart contract with corresponding data structures. On the right-hand side, attributes of the Commodity asset are specified with data types and names

a temperature value from a weather app and activating an internet of things (IoT) device through an IoT online platform in case of extreme conditions. In this category, the representation abstract from source code (column *r*) is either dialog-guided or flow-based. For example, IFTTT (WA-4) will present users with a series of dialogs prompting the involved web apps along with their actions and specific conditions.

An example offering more complex workflows in this area is n8n (WA-7), where a flow-based editor connects different services based on advanced application logic. Here, an application might retrieve documents from a spreadsheet web app and decide based on complex branching logic the processing with data operators; for instance, transformations of data columns and the calculation of formulas. Then, the processing in further apps might be triggered, e.g., adding entries to an SQL database or e-mailing the results with a mail application in case specified criteria are met.

Integration with blockchain transactions and smart contracts is supported in the surveyed approaches for the Ethereum blockchain in Zapier (WA-14) and Aurachain (WA-1), for the Amazon Managed Blockchain and Amazon Quantum Ledger Database in AWS, and for the Hyperledger Fabric blockchain in NodeRed (WA-8) and Aurachain (WA-1). Further blockchain integrations concern the support of crypto-currency data.

Smart contract development is supported primarily for Ethereum, Hyperledger Fabric, Hyperledger Sawtooth, Amazon Quantum Ledger Database in DAML (SC-1), Dappbuilder (SC-2), and Simba Chain (SC-3). In particular, Simba Chain supports smart contract design based on templates and a visual editor for Ethereum, Hyperledger Fabric, and others.

As an example, smart contract development in Simba Chain is shown in Fig. 9. In the editor, a data model consisting of assets and transactions is defined for a smart contract to be generated with data structures accordingly persisting data on the blockchain. In the example, the smart contract of the supply chain tracking application demonstrated before in Fig. 8 is defined in its data structures. From the app, it will store shipments with a certificate of origin and the commodities contained in it, specified with the displayed attributes such as global trade item number (GTIN), quantity, and weight data. Generated data structures allow for efficient data storage and will provide basic reading and writing functionality through setter and getter functions.

In comparison, DAML (SC-1) uses a text-based representation in the form of a domain-specific language that can specify complex smart contract functionality. The language uses textual descriptions with a notation that is mostly based on natural language elements, interpreted and deployed to a blockchain. The language encompasses primarily the specification of contract logic, e.g., triggering transactions based on

contractual conditions, the handling of tokens, e.g., sending out tokens if contractual conditions are met, and the definition of assets as abstract concepts represented by tokens. While this approach offers complex features due to the expressivity of the language, it is geared towards technical domain experts and developers as a low-code approach.

On the other side of the complexity spectrum, Dappbuilder (SC-2) offers smart contract creations through the instantiation of pre-defined templates for Ethereum, Polygon, and other blockchains. The approach is limited in applicability to the foreseen application areas that utilize standardized contracts modified according to the specification of the user, e.g., for issuing user-defined tokens on the Ethereum blockchain. For this purpose, a smart contract of the site is used based on the standardized Ethereum ERC-20 token implementation.

5.2.2 Platforms With 2nd degree integration

2nd degree blockchain integration has been found in 37 solutions intended for building websites, mobile or web apps, or forms, for workflow integration and automation, internal tools for companies, for data processing, and spreadsheet-based applications. Eight solutions also offer 1st degree blockchain integration. The integration features across the categories rely on another service supplying a direct integration for blockchain applications. Among the no-code or low-code applications, it is typical to integrate other services in the fashion of a composition, for example, creating an application in an app builder with data provided by an external service. Blockchain integration features, due to this capability, rely on other services for integration. Based on the integration paradigm, existing web services might be combined or access blockchain data.

Workflow integration and automation tools offer 2nd degree integration in this way. For example, workflows specified with cloud services such as with Amazon Web Services (AWS) (WA-2), or with Microsoft Azure (Azure) (WA-3) can define API calls in their flow-based editors to a platform with 1st degree integration, e.g., calling Zapier (WA-14) for the retrieval of Ethereum transactions. Similarly, workflow tools focused on integrating web apps and web services (WA-4 to WA-14) such as Pipefy (WA-9) might utilize Zapier and similar services in web-based workflows. In this way, most 2nd degree integration platforms offer the possibility of interacting with Ethereum smart contracts and transactions.

App builders and website builders provide further integration possibilities for mobile and web applications as well as websites. For instance, in Glide (AM-4), which can embed dialogs for smart contracts and transactions, in addition to integrating cryptocurrency data. In this way, applications composed of these components might be realized with workflow automation. Websites might also integrate dialogs from Glide (AM-4), e.g., e-commerce payments using cryptocurrencies or for enhancing websites with web3 functionality. Websites following the web3 concept [45] depend less on large, centralized, and potentially influential entities such as websites of well-known social networks. Instead, decentralized platforms and infrastructures host components of websites, e.g., using blockchain transactions made in a browser through the MetaMask extension or by decentralized data retrieved from Ethereum through services such as Zapier (WA-14). The integration paradigm can be applied and used in combination with platforms in the other categories concerning *data*, *form builders*, *internal tools*, and *spreadsheets*. For example, a transaction may be sent after the workflow has been started by another action such as entering data in a spreadsheet.

Spreadsheet tools may be used for building small applications with platforms such as Actiondesk (SP-1), AirTable (SP-2), or Rows (SP-3) by pre-defined components and multiple views that allow defining input fields and the processing of submitted data, e.g., calculating statistical measures of sales transactions on Ethereum retrieved through Zapier.

Form builders typically provide input fields for entering data with user-defined UI elements such as labels and buttons with the possibility of triggering external actions. For instance, retrieving corresponding transactions from Ethereum, and sending out results through notifications or e-mails. For example, Arengu might be used in this way (F-1).

Internal tools focus on company-internal tasks, encompassing enterprise tools for the automation of enterprise resource planning (ERP) or operational tasks. For example, by handling customer data and sales orders, managing human resources (HR), customer relationship management (CRM), creating dashboards or admin panels.For instance, using data from the Ethereum blockchain, tools such as Jet Admin (IN-2) might present a tabular structure and trigger an approval workflow for a back-office team.

Data processing tools permit the analysis of data with formulas or measures, e.g., of statistical nature, and offer predictive analytics features such as found in Levity (D-1) and Obviously AI (D-2). In this way, smart contract function calls might for example be evaluated regarding their frequency or sales volume. By permitting the integration of further data sources and the integration of other web applications, data analysis might involve the processing of newly appearing blockchain transactions. Such processing may include, for example, the storage of transaction data in spreadsheets, or the triggering of workflows. Integrations in this context can extend to enterprise-focused business process management when combined with platforms such as Process Street (WA-10), offering advanced functionality in combination with workflow engines.

Overall, the results show that the integration possibilities for the creation of websites or apps rely on few services such as Zapier (WA-14), predominantly found in the workflow

automation category. Typical integration features consist of access to blockchain transactions or cryptocurrency data. Further integration possibilities with APIs on a technical level are very common, however, they were not considered no-code or low-code when using Webhooks, Rest, other forms of HTTP requests, or API access based on technical parameters. For the development of smart contracts, few no-code platforms could be found in practice, with most solutions being low-code approaches using representations closer to source code where technical knowledge is required for development and blockchains.

6 Discussion

For discussing the results of our study we will first elaborate on the academic papers as shown in Tables 3 and 4. We characterize each topic with relevant papers found in the literature search. The topics were derived via the topic modeling technique of LDA and the subsequent manual refinement as has been described in Sects. 4.3 and 4.4. The selection of papers included in the discussion is based on all papers assigned to a particular topic with unanimous agreement by all raters (see Sect. 4.4 and Table 5) and discusses further papers from the literature search 4.3 that characterize each topic to illustrate the scope of approaches. This will permit us to give detailed insights into each of the proposed topics. Subsequently, we will draw the relations between the topics of the academic papers and the tools found in the analysis of low-code approaches. As a synthesis, we will be able to derive commonalities and differences in academic model-driven as well as industrial low- and no-code approaches for blockchain design and point to further research and development opportunities.

Process, workflow, choreography, and decision models: Model-driven design and implementation aspects of processes, workflows, choreographies, and decisions are supported in various approaches utilizing the blockchain at design-time or run-time. Concerning especially publications on process modeling and related subjects, the topic captures a substantial part of the overall literature on modeling and blockchains.

Centered on process and workflow models, most publications apply BPMN on the Ethereum blockchain and concern run-time aspects. Especially monitoring and execution are addressed by Weber et al. [263], following optimizations [102], and systems implementing execution such as Caterpillar [41, 159, 160] and further approaches focused on workflows, e.g., by Sturm et al. [231, 232] or based on annotations by Bagozi et al. [20]. For these execution approaches, BPMN is transformed to representations used for execution with a smart contract. Oftentimes, BPMN is transformed into an intermediate model closer to execution and then used as input for a generator program resulting in one or more smart contracts. The intermediate model has the role of specifying well-defined execution semantics, often relying on Petri Nets such as in [102, 163].

Business process management and workflow systems implementing execution consist of additional process or workflow engine components, listening to events emitted by the blockchain mostly through smart contracts as in Caterpillar and subsequent works [20, 160], usually with further blockchain-external components such as model repositories; only in some cases using decentralized repositories, e.g., based on distributed file systems as suggested by Bagozi et al. [20].

In few systems related to processes and workflows, approaches not relying on smart contract generation can be found. Here, processes and workflows are interpreted such as suggested by López-Pintado et al. [162], using workflow engines as also proposed by Falazi et al. [87, 88] and Sturm et al. [232]. Primarily static smart contracts, not changing over time, are used for the interpretation of process and workflow definitions. Also, combinations of static smart contracts and generated contracts could be found, e.g., proposed by Bagozi et al. [20] using smart contract templates, Härer suggesting a combination with multiple object- and participant-specific contracts [121] for business systems, and Klinger et al. also proposing partial generation of the contracts [142, 217].

While the approaches discussed so far rely primarily on BPMN and Ethereum with smart contracts, few process and workflow approaches also exist for other modeling languages and blockchain platforms. Not relying on smart contracts at all, Evermann and Kim [84] propose a workflow system based on blockchain transactions, where Petri Net representations of workflows are used to create transactions that trigger the execution on a workflow engine using a blockchain connector, possibly applicable also for bitcoin or other blockchains. Relying on statecharts, few works such as Nakamura et al. [185] can be found, where statecharts specified in SCXML are reduced and transformed to contracts on the Hyperledger Fabric blockchain for workflow execution with distributed node.js applications. SOM and BPMN are applied by Härer [121] for describing business systems consisting of structural views for distributed organizations in SOM and process behavior specified by BPMN collaboration diagrams, also concerning design-time aspects for collaborative modeling based on voting mechanisms together with the generic tracking of instances at run-time for monitoring.

DEMO is applied in few instances, e.g., by Loukil et al. suggesting a non-generative approach [164], where smart contracts are interpreted in a three-layered architecture and evaluated on Ethereum, and by Silva et al. [223] proposing DEMO for its actor transaction diagrams with further diagrams of transactions and UML classes as part of a comprehensive transformation method involving execution on the Hyperledger Fabric blockchain. Guerreiro et al. [114] apply

BPMN initially and suggest also DEMO for actor transaction diagrams that represent business transactions with a transformation implemented in Hyperledger Fabric smart contracts.

Concerning declarative modeling approaches, few works investigate decision models using DMN and case management by CMMN. For DMN, execution is proposed by Haarmann et al. [115, 117], where DMN provides decision rules based on formulas or decision tables that are encoded in smart contract logic for execution and demonstrated on the Ethereum blockchain. Concerning CMMN, Milani et al. [179] provide a comparison of BPMN with CMMN and point out their limitations regarding execution characteristics such as missing expressivity for completed tasks in CMMN.

Related to process choreographies, most works concern BPMN choreography diagrams with few approaches describing other modeling languages such as DCR graphs. Works in this area address the collaborative execution of process choreographies, where choreography tasks are executed by multiple participants exchanging messages. Based on BPMN and execution aspects, this is investigated by Ladleif et al. [146] and following approaches such as by Corradini et al. [61, 65], Sturm et al. [233], Corneli et al. [59], and Spalazzi et al. [229]. In BPMN choreographies, the control flow is restricted according to the choreography structure that is established by the tasks, relationships, and specialized gateways and events, effectively defining permitted patterns of message exchanges. Most approaches generate smart contracts such as Corradini et al. [61], possibly with the non-generative use of static components such as by Ladleif et al. [146]. In addition to the Ethereum-based approaches, also a multi-chain implementation has been proposed by Ladleif et al. [145].

Choreographies based on languages beside BPMN appear in comparatively few works based on DCR as addressed by Madsen et al. [166] and Henry et al. [130] in methods allowing for execution on the Ethereum blockchain. Provided an external execution engine exists, control flows between collaborative tasks of multiple participants are restricted to the edges defined by the DCR graphs in these approaches.

Further domain-specific notations exist in few works, not applying standardized modeling languages. For workflows, JSON is used by Bore et al. [38] for execution in a workflow system connected to Hyperledger Fabric. Another domain-specific notation using transaction concepts derived from Ethereum for business processes is suggested by Boychenko and Gavrikov [40] by a UI prototype without implementation.

Further specializations for the body of literature discussed for process and workflow approaches exist in addition, primarily concerning run-time aspects and optimizations. For example, on traceability and monitoring by Di Ciccio et al. [55], confidential execution such as in Carminati et al. [48], trust, e.g., Bagozi et al. [21], and flexibility aspects for instance in López-Pintado et al. [163], as well as BPMN-specific concepts, e.g., with focus on time and events by Abid et al. [2] and Naha et al. [247].

Application development: The development of blockchain-based applications using models and model-driven engineering has been the subject of many retrieved publications. Thus, the range of methods found for this topic comprises both conceptual and executable, code-generating approaches and spans from base languages such as AOM, e.g., [153], BPMN and DMN, e.g., [192], DEMO, e.g., [14], ER, e.g., [100], OCL, e.g., [235], OWL [28] to RuleML, e.g., [256], and UML, e.g., [101, 139, 168, 237], as well as a large number of blockchain domain-specific languages, e.g., [39, 99, 120, 158, 210, 236, 271], and visual programming paradigms such as Blockly, e.g., [167, 178, 265]. In terms of the targeted blockchain platforms for application development approaches, we could find a strong preference for focusing on the Ethereum platform, some also directed towards Hyperledger as a permissioned platform, e.g., [100, 139, 178], and only few with an explicit support of multiple platforms, e.g., [119, 120, 128, 206, 235]. Of particular interest from the perspective of modeling are approaches that extend existing languages with blockchain-specific concepts, e.g., by using stereotypes or profiles, as they may permit a gradual evolution of existing models towards blockchain-specific ones. This has been applied for example to UML, e.g., [128, 168, 170] or NodeRed [210]. Further approaches proposing UML profiles explicitly will be discussed below under the topic of UML Modeling.

Formal aspects and verification: The use of formal methods in the context of blockchain-based applications and especially for smart contracts is a strongly researched topic [246]. In particular, the formal verification of smart contracts can ensure their correctness and thus avoid errors that cannot be corrected afterwards due to the immutable nature of blockchains [183]. The approaches we found in our study mostly revert to some custom notation, which is the reason why we classified them under domain-specific language. They are often based on a custom-developed variant of state machines/diagrams, automata, or tool-specific extensions such as stateflows in Matlab/Simulink—e.g., [11, 172, 177]—but mostly do not adhere to the UML statechart specification. The exception being [216] where statecharts expressed in SCXML are generated during formal model checking and used as a basis for the generation of smart contract code on the Hyperledger platform. Further languages that have been used for formal verifications include for example Petri Nets for the visual modeling, simulation, and verification of smart contracts as a basis for code generation [277] and RDF as a foundation for representing knowledge about smart contract vulnerabilities and verification rules that are checked prior to code generation [201]. Although not directed to formal verification per se, also

approaches based on logic-based languages such as DatalogMTL are considered here, which can be used for formal modeling of smart contracts and according code generation [190]. In addition, approaches can be found that revert to UML profiles for blockchain applications with explicit verification rules described in programming code [109], thus providing a link to the application development topic.

Legal aspects, rules, and languages: Due to the slightly misleading term "smart *contract*", it is frequently assumed that this concept corresponds fully to a legal contract. However, as has recently been elaborated, four types of relations between smart contracts and legal agreements need to be distinguished [83, p.25]: (1) the smart contract stands for mere code without a legal agreement, (2) the smart contract acts as a tool to execute a legal agreement, whereby the latter is maintained off-chain, (3) the smart contract constitutes a legal agreement by itself, e.g., to express an offer or acceptance, or (4) the smart contract and a legal agreement are merged and they both exist simultaneously on-chain and off-chain, whereby it needs to be stated whether the agreement is to be treated on-chain or off-chain. These preliminaries need to be considered when regarding papers that treat *legal aspects* in the context of blockchains. In particular, approaches of the first type (1) are discussed under the topic Application Development. Further, also the terms *rules* and *languages* were identified as part of this topic in the course of the content-based analysis using LDA. We thus group here together papers that incorporate rule-based approaches or that aim for the development of new languages in the context of blockchains, thereby reverting also to modeling or model-driven engineering.

In terms of legal aspects, we thus identified papers that develop, for example, a smart legal contract markup language, dedicated new languages, or transformations from existing legal or financial contract languages into blockchain-based smart contract languages thus arriving at legally-binding smart contracts, i.e., where the smart contract constitutes a legal agreement [51, 80, 148, 267]. Another direction is to use smart contracts for partially automating the execution of legal contracts as done in the MDE-based Beagle approach where the legal agreement is kept off-chain [251]. Furthermore, declarative approaches have been proposed for representing contractual terms for deriving smart contracts on blockchain platforms, e.g., using RuleML as base language [71, 72]. In terms of dedicated languages for smart contract development there is some overlap with the topic of Application Development discussed above. Thus, approaches such as Das Contract [226], SmaCoNat [207], SPESC [125], or CML [270] also mention their relation to paper-based legal contracts and may be used for legal purposes but focus more on the development of blockchain applications. We also found papers on the conceptual level, e.g., for unifying the notion of legal smart contracts from a conceptual modeling perspective [144] or for the description of legal concepts in smart contracts using formal ontologies based on Petri Nets [78].

Ontology: Various forms of ontologies in different languages and formats are used for designing blockchain-based applications, as seen, for example, in the above topic where ontologies mainly take on a supportive role. Under this topic we thus classify those approaches that put the focus on ontologies for blockchain applications and which regard ontologies not only as an additional means of representation. This includes for example the use of ontologies in a way similar to reference models, i.e., for achieving a common understanding of the contained concepts as proposed in [70] for explaining the nature of blockchain transactions and for providing a taxonomy of the blockchain ecosystem by using UML or in [69] with the focus on smart contracts. This direction can be further extended by reverting to ontology languages such as OWL and SWRL which permit to formalize blockchain-related concepts such as NFTs (Non-Fungible Tokens), DAOs (Decentralized Autonomous Organizations), or oracles from different perspectives and reason about them using rules or queries [26, 29]. Apart from general reasoning about blockchain concepts, ontologies have been used to reason about specific blockchain technologies, e.g., the Solidity language for Ethereum smart contracts, which permits, for example, to derive statistics about particular concepts used in deployed contracts [44].

UML modeling: Based on the topics suggested by the LDA, we decided to establish a separate topic for approaches that put an explicit focus on UML. Similar to what we noted for the topic of ontologies, UML has already been covered in several approaches as a means of representation and a basis for execution. Here we thus focus on approaches that put UML at the center. This includes, for example, approaches that revert to UML use case and UML sequence diagrams in combination with i* for understanding the goals and design of blockchain-based software applications [258, 259]. As had already been mentioned above, the extension of UML through blockchain-specific profiles or stereotypes has been proposed in several papers. Górski and Bednarski thus suggest a UML profile for smart contracts by providing stereotypes for states, contracts, verification rules in smart contracts, and flows for representing the communication of nodes and show how this can be translated into an implementation on the Corda platform [110]. The same authors further proposed a UML profile for the deployment of distributed ledger applications. For that purpose they added stereotypes to UML deployment diagrams for distinguishing, for example, between DLT nodes, oracle nodes and nodes specific to the targeted Corda platform [111–113]. Standard UML models such as statecharts may also be used for engaging in application development via model-driven engineering. Jurgelaitis et al. have shown for example, how

UML class diagrams and statecharts may be used for representing platform-independent and platform-specific models for generating code in the Solidity language for the Ethereum platform [137].

Business modeling: Under the topic of business modeling we identified only approaches that are positioned on the conceptual level. Although the generation of artifacts from models in this area has been demonstrated—e.g., for generating business plans from business model canvas models [268]—we have not found approaches in our study that apply such techniques for the design of blockchain-based applications so far. Of particular interest in this context are approaches for identifying new business opportunities through the use of modeling approaches, e.g., by using languages such as e^3 value [198, 203], e^3 value in combination with BPMN [105], or ArchiMate [135]. Thereby, the latter may also be integrated with technical aspects of the underlying enterprise architecture such as shown in [66]. Further, it has been shown how multi-agent organizational modeling can be applied for studying and simulating blockchain-based systems [211], whereby UML has been chosen to describe the approach.

Reference models: The creation and application of reference models has been widely studied in multiple domains. Reference models can be characterized as conceptual models that serve as model blueprints in a certain domain [269] and especially for the development of information systems [91]. For the field of blockchains, reference models have been proposed for describing the underlying technologies and their procedures as such, e.g., using a combination of ArchiMate, BPMN, and UML [82], using custom notations for specific blockchain platforms such as Hyperledger [37], or using UML for describing the conceptual schema of Ethereum [194].

Supply chain: The application of blockchain technologies for tracing products over large logistic networks and across different cooperating parties has gained considerable interest both in academic research as well as in practice [106]. Thereby, the digitalization of the physical workflow of documents and transactions between the actors in a supply chain such as shipment companies, ports, airports, etc. has the potential to reduce transaction costs and enable even new types of business services based on the availability of additional data. This is especially due to the use of blockchains in practice for this area as recently exemplified through the TradeLens platform [136]. In this context, we found approaches that focus specifically on the area of supply chains and that provide model-based approaches. This includes the use of established modeling approaches such as i* and BPMN, e.g., for representing the interaction of business actors in the realm of supply chains [118], or the development of domain-specific modeling languages and profiles as well as dedicated blockchains for supply chain applications, as e.g., in [15, 31, 32, 222].

Industrial low-code and no-code approaches: The review of industrial low-code and no-code approaches concluded with 47 software platforms. In this discussion, the platforms described in Sect. 5.2 and Table 6 are summarized for their main characteristics and limitations, leading to a concluding discussion in comparison to academic approaches for model-driven engineering. In comparison to academic approaches, a broad comparison is given in Table 7.

The reviewed platforms are available as online platforms following the software-as-a-service paradigm, partially with additional offline components. They can provide technical solutions for practitioners ranging from the automation of business tasks to workflow integration and software engineering.

From the review, platform tests, and implementations for evaluation (c.f. Sect. 5.2), a general finding is the high maturity of industrial platforms in the form of products. In particular, the high usability and the availability of a broad range of interfaces for cloud-based and blockchain integrations, and the possibility of cross-platform development. 18 platforms allowed for 1st degree integration with blockchains, primarily with Ethereum, Hyperledger Fabric and Amazon Quantum Ledger Database, while 37 platforms provided 2nd degree integration by relying on other platforms or services. This result is similar to the blockchain platforms identified for the academic approaches in Table 3.

For blockchain application development, 1st degree integrations allow reading of blockchain data as well as submitting transactions to major blockchain networks in addition to specific functionality regarding templates for decentralized applications and cryptocurrency-related data integrations. Using the principle of composition, blockchain applications can utilize applications and web services as well as platform-as-a-service and infrastructure-as-a-service offerings from cloud providers, possibly followed by data processing and analytics. Blockchain integration is therefore applicable primarily with categories related to smart contracts, workflow integration and automation, and data (Table 6). Furthermore, data transfers, triggering of events, or presentation in application UIs or dashboards may be realized. Despite these features, however, limitations related to the abstraction from source code, implementation trade-offs, and technical knowledge remain.

The representations abstract from source code such as flow-based editors (c.f. Fig. 8) or graph-based data model editors (c.f. Fig. 9) provide meaningful abstraction, ease-of-use, and, at the same time, hide implementation complexity. The insight into the execution behavior is limited in most platforms, where few platforms allow for limited source code access, e.g., Outsystems and Mendix. This results in limitations primarily of auditability, transparency, portability, and

migration. For instance, in cases where execution behavior requires inspections due to audits, bugs or other unintended behavior, access to source code is required.

Also related to abstractions is the technical knowledge required for users. Low-code development environments such as in Outsystems or Mendix provide representations close to source code and tend to be better suited for developers or technical domain experts, possibly engaging in application development or supporting it. No-code environments might be used by non-technical experts or end users, sometimes called citizen developers, for their abstractions beyond the level of source code, e.g., guiding the user through dialogues. Here, business tasks involving multiple web applications might be automated, however, the development of complex applications involving blockchains can at most be supported, e.g., in discussing implementations with technical experts and domain experts. For both low-code and no-code platforms, visualization and potentially better understanding of domain knowledge are general benefits, limited by technical knowledge required for developing with the platforms, the understanding of execution behavior, especially in case of unintended behavior, and the possibility of evaluating implementation trade-offs.

Comparison of academic and industrial approaches: When comparing the results of the analysis of academic approaches and those of industrial low-code and no-code approaches we can identify the following commonalities and differences. Although industrial approaches typically do not position their platforms under the terms *model-driven* or *model-based*, the representations used in some of them directly correspond to these terms as used in the academic publications—see for example the illustrations of the Outsystems and the Simba chain platforms in Figs. 8 and 9.

While academic approaches have discussed the design of blockchain-based applications from a multitude of perspectives, including for example business, legal, organizational, deployment, and coding aspects, the industrial low-code and no-code platforms seem rather limited in terms of their scope. For example, none of the found low-code or no-code approaches provide representations to elaborate on business aspects, the definition of goals or the consideration of legal and compliance aspects. In these areas industrial approaches could consider in the future the methods and tools proposed by academia. This concerns in particular also the integration of different methods as has been broadly discussed in enterprise modeling in academia.

On the other hand, the industrial platforms all have a high technical maturity, especially regarding ease of use, integration, and, in some platforms, features for scalability. These features are not directly comparable to aspects of academic approaches which often are only partially implemented, focus on feasibility demonstrations, and generally do not offer features found in industrial software products.

Since most platforms are web-based, they tend to integrate also the hosting of cloud-based application components or services that can be scaled by the platform or through integrated services from cloud providers. Especially when using *serverless* architectures from cloud providers such as AWS, scaling can extend to further infrastructure resources without requiring changes on the application level. This is of course something that cannot be reached by academic platforms which can typically only provide a prototypical implementation, which is ideally made accessible as open-source.

A further interesting aspect where academic and industrial approaches differ at this point in time are proposals for domain- or task-specific solutions. As could be found in the academic discussions on blockchain design approaches in the area of supply chains, first approaches dedicated to this specialized field exist. In contrast, industrial approaches so far seem to rest on the general level, at least when it comes to 1st degree integration platforms. However, platforms such as Budibase (IN-1) or Jet Admin (IN-2) would offer at least the possibility of 2nd degree integration here.

Limitations of the study: Despite the extensive review of literature sources across many different outlets and the inspection of a large number of low-code and no-code platforms our study is of course not without limitations. These concern both the retrieval of literature, the selection of relevant sources, as well as their interpretation. Regarding the retrieval of sources we may have missed publications that could only be found using multiple recursions. Here, we set a cut-off in the backward and forward search due to our limited manual processing capacities. Bigger author teams with additional resources may thus contribute additional results. The same applies for the retrieval of industrial low-code and no-code platforms and tools. Concerning the determination of the relevance of publications for the investigated topic and the derived research questions, a certain subjectivity cannot be avoided. We aimed to mitigate this as much as possible by going through multiple iterations and in-depth discussions between the authors. Finally, the content-based analysis of the literature sources also has a subjective component. As a mitigation we reverted to a computational approach for a more objective proposal of potential topics and a later manual refinement. However, as has already been mentioned in Sect. 4.3, also the used LDA approach requires a subjective selection of parameters and could easily lead to different results in case of variations. We are however confident that the provision of all used datasets and analysis parameters makes the used methods and processes transparent and replicable—see "Appendix A".

Table 7 Primary characteristics of academic and industrial approaches for blockchain application development

Blockchain application development	Academic model-driven engineering approaches	Industrial low-code and no-code approaches
Resulting artifacts	Focus on blockchain and platform models concepts, design languages, frameworks, formal proofs, ontologies, transformation	Focus on blockchain software products, cloud- and web-based software (SaaS paradigm), inttegrated with platforms
Scope of activities	Model-driven support for one or many blockchain platform development and engineering aspects, model-based	Providing technical solutions for automation integration, focus on tasks or workflows, inter-active low- and no-code editors
Abstraction level	Arbitrary abstraction level, holistic or specific aspects of business, application technical levels, possibility of integration and alignment across levels	Technical representations, mostly within one level of abstraction, focused on application and technical levels
Target group	Primarily researchers, professional software or requirements engineers, limited accessibility for domain experts and citizen developers	Primarily domain experts, industry professionals, web and software developers, citizen developers
Implementation maturity	Often low, focus on concepts, proof-of-concept software, prototypes, demonstrators	Often high, focus on products, user interfaces integration through APIs and cloud platforms user experience
Integration	Predominantly stand-alone approaches without integration capabilities	Predominantly cloud- and web-based solutions with integration capabilities and APIs

SaaS software-as-a-service, *API* application programming interface

7 Conclusion and research opportunities

In this paper we reviewed academic model-driven engineering approaches and industrial low-code and no-code platforms for supporting the development of blockchain-based applications. We have done this along four research questions, restated in the following:

RQ1: *Which academic modeling approaches have been developed until today for designing blockchain-based applications?*

We have identified 177 academic publications (excluding preprint versions that have been published) from various fields regarding model-driven engineering and development of blockchain-based applications. Further we identified the modeling methods used, as well as the blockchain technologies of interest in academia.

RQ2: *Which low-code and no-code platforms permit the realization of blockchain-based applications?*

Following a rigorous review process, 47 low-code and no-code platforms have been identified. We have found that these realize blockchain support either directly, as feature implemented in the product, or indirectly via integration facilities.

RQ3: *What are the predominant characteristics and areas of academic modeling approaches as well as low-code and no-code platforms?*

Academic approaches display an effort towards a strong conceptualization of the blockchain-domain with a focus on various types of models as central artifact. In contrast, low-code and no-code platforms typically are less concerned with model-representations, or hide these from the user. Instead, these platforms focus on software products, oftentimes as part of a cloud-based solution. In terms of technical maturity, most academic approaches do not go beyond a prototypical realization and instead place more weight on conceptual aspects. In this regard, industry solutions typically feature a higher maturity and offer greater flexibility for an integration with other products.

RQ4: *What are future research opportunities not realized today by academic approaches and low-code or no-code platforms?*

Based on the insights we gained through our study we can derive the following opportunities for further research in the design of blockchain-based applications in a. model-driven engineering, b. industrial low-code and no-code platforms, and c. combinations and interfaces between academic and industrial approaches.

Despite the large number of approaches proposed in the academic papers and the multitude of different languages used for supporting the design of blockchain-based applications we noticed a shortage of holistic approaches covering both business as well as technical aspects. For example, the implementation of blockchain solutions not only requires a sound and sustainable business model but also ways for translating this into technical architectures and possibly even to the level of code generation. At the moment this could only be achieved using a combination of several approaches, similar to the 2nd degree integration found in the review of industrial tools and platforms. Further, only few approaches exist so far that target explicitly the design of blockchain-oriented business models. Though existing languages such as e^3 value or parts of ArchiMate can and have been shown to be used for this purpose, languages with explicit stereotypes for this purpose have not yet been proposed. Also for other modeling aspects such as enterprise architecture, business processes and workflows, and the technical realization, further profiles and stereotypes could be proposed to ease the usage of model-based approaches.

The review of industrial low-code and no-code platforms showed the advanced state of these platforms today compared to their counterparts found in early endeavors in the 1990–ies. As has already been mentioned in the comparison above, these platforms however lack methods for adding more conceptual aspects and relations to the business side. In this respect, they could learn from the academic approaches and possibly even academia could contribute here in the form of plugins or extensions of existing industrial platforms. Further development opportunities for the industrial platforms could be identified in regard to better support for debugging and error handling, where still considerable technical knowledge is required by users.

Finally, we can derive opportunities for joining academic and industrial approaches. Apart from the mentioned extensions of platforms on either side, it may be worthwhile establishing interfaces between academic and industrial platforms. For example, academic platforms could provide interfaces to profit from advances in formal verification of smart contracts in industrial platforms and low-code and no-code platforms could provide APIs for enabling interaction with academic platforms, e.g., on the business and conceptual level.

Appendix A: Datasets of the review process

The bibliographies of the document corpora at various stages of the review process corresponding to Fig. 2 are available online [68]. In particular, we provide reference lists for all documents collected after the steps (S-2), (S-4), (S-7) and (S-8).

Acknowledgements This work was supported by the Swiss National Science Foundation project Domain-Specific Conceptual Modeling for Distributed Ledger Technologies [196889].

Funding Open access funding provided by University of Fribourg

References

1. Abbas, M., Rashid, M., Azam, F., Rasheed, Y., Anwar, M.W., Humdani, M.: A model-driven framework for security labs using blockchain methodology. In: IEEE International Systems Conference, SysCon 2021, Vancouver, BC, Canada, April 15–May 15, 2021, pp. 1–7. IEEE (2021). https://doi.org/10.1109/SysCon48628.2021.9447125
2. Abid, A., Cheikhrouhou, S., Jmaiel, M.: Modelling and executing time-aware processes in trustless blockchain environment. In: Risks and Security of Internet and Systems, 14th International Conference, CRiSIS 2019, Hammamet, Tunisia, October 29–31, 2019, Proceedings. Lecture Notes in Computer Science, vol. 12026, pp. 325–341. Springer (2019). https://doi.org/10.1007/978-3-030-41568-6_21
3. Ait Hsain, Y., Laaz, N., Mbarki, S.: Ethereum's smart contracts construction and development using model driven engineering technologies: a review. In: The 2nd International Workshop on the Advancements in Model Driven Engineering (AMDE) March 23–26, 2021, Warsaw, Poland. Procedia Computer Science, vol. 184, pp. 785–790. Elsevier (2021). https://doi.org/10.1016/j.procs.2021.03.097
4. Alam, M.T., Chowdhury, S., Halder, R., Maiti, A.: Blockchain domain-specific languages: survey, classification, and comparison. In: 2021 IEEE International Conference on Blockchain, Blockchain 2021, Melbourne, Australia, December 6–8, 2021, pp. 499–504. IEEE (2021). https://doi.org/10.1109/Blockchain53845.2021.00076
5. Allouche, M., Mitrea, M., Moreaux, A., Kim, S.: Automatic smart contract generation for internet of media things. ICT Express **7**(3), 274–277 (2021). https://doi.org/10.1016/j.icte.2021.08.009
6. Almakhour, M., Sliman, L., Samhat, A.E., Mellouk, A.: Verification of smart contracts: a survey. Pervasive Mob. Comput. **67**, 101227 (2020). https://doi.org/10.1016/j.pmcj.2020.101227

7. Alves, P.H.C., Paskin, R., Frajhof, I.Z., Miranda, Y.R., Jardim, J.G., Cardoso, J.J.B., Tress, E.H.H., Ferreira da Cunha, R., Nasser, R., Robichez, G.: Exploring blockchain technology to improve multi-party relationship in business process management systems. In: Proceedings of the 22nd International Conference on Enterprise Information Systems, ICEIS 2020, Prague, Czech Republic, May 5–7, 2020, Vol. 2, pp. 817–825. ScitePress (2020). https://doi.org/10.5220/0009565108170825
8. Amaral De Sousa, V., Burnay, C.: MDE4BBIS: a framework to incorporate model-driven engineering in the development of blockchain-based information systems. In: Third International Conference on Blockchain Computing and Applications, BCCA 2021, Tartu, Estonia, November 15–17, 2021, pp. 195–200. IEEE (2021). https://doi.org/10.1109/BCCA53669.2021.9657015
9. Amaral De Sousa, V., Burnay, C., Snoeck, M.: B-MERODE: a model-driven engineering and artifact-centric approach to generate blockchain-based information systems. In: Advanced Information Systems Engineering—32nd International Conference, CAiSE 2020, Grenoble, France, June 8–12, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12127, pp. 117–133. Springer (2020). https://doi.org/10.1007/978-3-030-49435-3_8
10. Amato, F., Cozzolino, G., Moscato, F., Moscato, V., Xhafa, F.: A model for verification and validation of law compliance of smart contracts in IoT environment. IEEE Trans. Ind. Inform. **17**(11), 7752–7759 (2021). https://doi.org/10.1109/TII.2021.3057595
11. Andrychowicz, M., Dziembowski, S., Malinowski, D., Mazurek, L.: Modeling Bitcoin contracts by timed automata. In: Formal Modeling and Analysis of Timed Systems—12th International Conference, FORMATS 2014, Florence, Italy, September 8–10, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8711, pp. 7–22. Springer (2014). https://doi.org/10.1007/978-3-319-10512-3_2
12. Antonopoulos, A.M., Wood, G.: Mastering Ethereum: Building Smart Contracts and DApps. O'Reilly Media, Sebastopol (2018)
13. Aparício, M., Guerreiro, S., Sousa, P.: Automated DEMO action model implementation using blockchain smart contracts. In: Proceedings of the 12th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, IC3K 2020, Volume 2: KEOD, Budapest, Hungary, November 2–4, 2020, pp. 283–290. ScitePress (2020a). https://doi.org/10.5220/0010147602830290
14. Aparício, M., Guerreiro, S., Sousa, P.: Towards an automated DEMO action model implementation using blockchain smart contracts. In: Proceedings of the 22nd International Conference on Enterprise Information Systems, ICEIS 2020, Prague, Czech Republic, May 5–7, 2020, Vol. 2, pp. 762–769. ScitePress (2020b). https://doi.org/10.5220/0009417907620769
15. Asawa, K., Kukreja, S., Gondkar, R.: An NCDP for developing a blockchain based dynamic supply chain management with auto-generation of smart contract. In: 26th International Conference on Automation and Computing, ICAC 2021, Portsmouth, United Kingdom, September 2–4, 2021, pp. 1–6. IEEE (2021). https://doi.org/10.23919/ICAC50006.2021.9594235
16. Astigarraga, T., Chen, X., Chen, Y., Gu, J., Hull, R., Jiao, L., Li, Y., Novotný, P.: Empowering business-level blockchain users with a rules framework for smart contracts. In: Service-Oriented Computing—16th International Conference, ICSOC 2018, Hangzhou, China, November 12–15, 2018, Proceedings. Lecture Notes in Computer Science, vol. 11236, pp. 111–128. Springer (2018). https://doi.org/10.1007/978-3-030-03596-9_8
17. Azzopardi, S., Colombo, C., Pace, G.J.: Model-based static and runtime verification for Ethereum smart contracts. In: Model-Driven Engineering and Software Development—8th International Conference, MODELSWARD 2020, Valletta, Malta, February 25–27, 2020, Revised Selected Papers. Communications in Computer and Information Science, vol. 1361, pp. 323–348. Springer (2020). https://doi.org/10.1007/978-3-030-67445-8_14
18. Azzopardi, S., Ellul, J., Pace, G.J.: Runtime monitoring processes across blockchains. In: Fundamentals of Software Engineering—9th International Conference, FSEN 2021, Virtual Event, May 19–21, 2021, Revised Selected Papers. Lecture Notes in Computer Science, vol. 12818, pp. 142–156. Springer (2021). https://doi.org/10.1007/978-3-030-89247-0_10
19. Babkin, E., Komleva, N.: Model-driven liaison of organization modeling approaches and blockchain platforms. In: Advances in Enterprise Engineering XIII—9th Enterprise Engineering Working Conference, EEWC 2019, Lisbon, Portugal, May 20–24, 2019, Revised Papers. Lecture Notes in Business Information Processing, vol. 374, pp. 167–186. Springer (2019). https://doi.org/10.1007/978-3-030-37933-9_11
20. Bagozi, A., Bianchini, D., Antonellis, V.D., Garda, M., Melchiori, M.: A three-layered approach for designing smart contracts in collaborative processes. In: On the Move to Meaningful Internet Systems: OTM 2019 Conferences—Confederated International Conferences: CoopIS, ODBASE, C&TC 2019, Rhodes, Greece, October 21–25, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11877, pp. 440–457. Springer (2019). https://doi.org/10.1007/978-3-030-33246-4_28
21. Bagozi, A., Bianchini, D., De Antonellis, V., Garda, M., Melchiori, M.: A blockchain-based approach for trust management in collaborative business processes. In: Web Information Systems Engineering—WISE 2021—22nd International Conference on Web Information Systems Engineering, WISE 2021, Melbourne, VIC, Australia, October 26–29, 2021, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13080, pp. 59–67. Springer (2021). https://doi.org/10.1007/978-3-030-90888-1_5
22. Bai, X., Cheng, Z., Duan, Z., Hu, K.: Formal modeling and verification of smart contracts. In: Proceedings of the 7th International Conference on Software and Computer Applications, ICSCA 2018, Kuantan, Malaysia, February 08–10, 2018, pp. 322–326. ACM (2018). https://doi.org/10.1145/3185089.3185138
23. Baresi, L., Quattrocchi, G., Tamburri, D.A., Terracciano, L.: A declarative modelling framework for the deployment and management of blockchain applications. CoRR (2022). https://doi.org/10.48550/ARXIV.2209.05092
24. Barisic, A., Zhu, E., Mallet, F.: Model-driven approach for the design of multi-chain smart contracts. In: 3rd Conference on Blockchain Research and Applications for Innovative Networks and Services, BRAINS 2021, Paris, France, September 27–30, 2021, pp. 37–38. IEEE (2021). https://doi.org/10.1109/BRAINS52497.2021.9569809
25. Barnett, J., Akolkar, R., Auburn, R.J., Bodell, M., Burnett, D.C., Carter, J., McGlashan, S., Lager, T., Helbing, M., Hosn, R., Raman, T.V., Reifenrath, K., Rosenthal, N., Roxendal, J.: State Chart XML (SCXML): State Machine Notation for Control Abstraction (2015). https://www.w3.org/TR/scxml/
26. Bella, G., Cantone, D., Longo, C., Nicolosi Asmundo, M., Santamaria, D.F.: Blockchains through ontologies: The case study of the Ethereum ERC721 standard in OASIS. In: Intelligent Distributed Computing XIV, 14th International Symposium on Intelligent Distributed Computing, IDC 2021, Virtual Event, 16–18 September 2021. Studies in Computational Intelligence, vol. 1026, pp. 249–259. Springer (2021). https://doi.org/10.1007/978-3-030-96627-0_23
27. Ben Slama Souei, W., El Hog, C., Sliman, L., Ben Djemaa, R., Ben Amor, I.A.: Towards a uniform description language for smart contract. In: 30th IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE 2021, Bayonne, France, October 27–29, 2021, pp. 57–62. IEEE (2021). https://doi.org/10.1109/WETICE53228.2021.00022

28. Besançon, L., Ghodous, P., Gelas, J.-P., da Silva, C.F.: Modelling of decentralised blockchain applications development. In: The 2020 International Conference on High Performance Computing and Simulation (HPCS 2020). HPCS (2021)
29. Besançon, L., da Silva, C.F., Ghodous, P., Gelas, J.: A blockchain ontology for Dapps development. IEEE Access **10**, 49905–49933 (2022). https://doi.org/10.1109/ACCESS.2022.3173313
30. Biryukov, A., Khovratovich, D., Tikhomirov, S.: Findel: secure derivative contracts for Ethereum. In: Financial Cryptography and Data Security—FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA, Sliema, Malta, April 7, 2017, Revised Selected Papers. Lecture Notes in Computer Science, vol. 10323, pp. 453–467. Springer (2017). https://doi.org/10.1007/978-3-319-70278-0_28
31. Bistarelli, S., Faloci, F., Mori, P.: *.chain: Automatic coding of smart contracts and user interfaces for supply chains. In: Third International Conference on Blockchain Computing and Applications, BCCA 2021, Tartu, Estonia, November 15–17, 2021, pp. 164–171. IEEE (2021a). https://doi.org/10.1109/BCCA53669.2021.9656987
32. Bistarelli, S., Faloci, F., Mori, P.: Towards a graphical DSL for tracing supply chains on blockchain. In: Euro-Par 2021: Parallel Processing Workshops—Euro-Par 2021 International Workshops, Lisbon, Portugal, August 30–31, 2021, Revised Selected Papers. Lecture Notes in Computer Science, vol. 13098, pp. 219–229. Springer (2021b). https://doi.org/10.1007/978-3-031-06156-1_18
33. Bistarelli, S., Faloci, F., Mori, P., Taticchi, C.: Olive oil as case study for the—chain platform. In: Proceedings of the 4th Workshop on Distributed Ledger Technology Co-located with the Italian Conference on Cybersecurity 2022 (ITASEC 2022), Rome, Italy, June 20, 2022. CEUR Workshop Proceedings, vol. 3166, pp. 94–102. CEUR (2022)
34. Blei, D.M.: Probabilistic topic models. Commun. ACM **55**(4), 77–84 (2012). https://doi.org/10.1145/2133806.2133826
35. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
36. Bock, A.C., Frank, U.: Low-code platform. Bus. Inf. Syst. Eng. **63**(6), 733–740 (2021)
37. Bollen, P.: A conceptual model of the blockchain. In: On the Move to Meaningful Internet Systems: OTM 2019 Workshops—Confederated International Workshops: EI2N, FBM, ICSP, Meta4eS and SIAnA 2019, Rhodes, Greece, October 21–25, 2019, Revised Selected Papers. Lecture Notes in Computer Science, vol. 11878, pp. 117–126. Springer (2019). https://doi.org/10.1007/978-3-030-40907-4_12
38. Bore, N., Kinai, A., Mutahi, J., Kaguma, D., Otieno, F., Remy, S.L., Weldemariam, K.: On using blockchain based workflows. In: IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2019, Seoul, Korea (South), May 14–17, 2019, pp. 112–116. IEEE (2019). https://doi.org/10.1109/BLOC.2019.8751446
39. Boubeta-Puig, J., Rosa-Bilbao, J., Mendling, J.: CEPchain: a graphical model-driven solution for integrating complex event processing and blockchain. Expert Syst. Appl. **184**, 115578 (2021). https://doi.org/10.1016/j.eswa.2021.115578
40. Boychenko, O.V., Gavrikov, I.V.: Potential applications of smart contract technology in corporate business processes. In: Distributed Computer and Communication Networks, vol. 1141, pp. 612–624. Springer (2019). https://doi.org/10.1007/978-3-030-36625-4_49
41. Brahem, A., Messai, N., Sam, Y., Bhiri, S., Devogele, T., Gaaloul, W.: Running transactional business processes with blockchain's smart contracts. In: 2020 IEEE International Conference on Web Services, ICWS 2020, Beijing, China, October 19–23, 2020, pp. 89–93. IEEE (2020). https://doi.org/10.1109/ICWS49710.2020.00019
42. Brambilla, M., Cabot, J., Wimmer, M.: Model-Driven Software Engineering in Practice, 2nd edn. Morgan & Claypool, San Rafael (2017)
43. Brandt, S., Güzel Kalaycı, E., Kontchakov, R., Ryzhikov, V., Xiao, G., Zakharyaschev, M.: Ontology-based data access with a horn fragment of metric temporal logic. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31(1). AAAI (2017). https://doi.org/10.1609/aaai.v31i1.10696
44. Cano-Benito, J., Cimmino, A., García-Castro, R.: Toward the ontological modeling of smart contracts: a solidity use case. IEEE Access **9**, 140156–140172 (2021). https://doi.org/10.1109/ACCESS.2021.3115577
45. Cao, L.: Decentralized AI: edge intelligence and smart blockchain, Metaverse, Web3, and DeSci. IEEE Intell. Syst. **37**(3), 6–19 (2022). https://doi.org/10.1109/MIS.2022.3181504
46. Cardano Foundation: Cardano (2022). https://cardano.org/. Accessed 19 Oct 2022
47. Cardoso, J., Barros, A., May, N., Kylau, U.: Towards a unified service description language for the internet of services: requirements and first developments. In: 2010 IEEE International Conference on Services Computing, pp. 602–609. IEEE (2010). https://doi.org/10.1109/SCC.2010.93
48. Carminati, B., Rondanini, C., Ferrari, E.: Confidential business process execution on blockchain. In: 2018 IEEE International Conference on Web Services, ICWS 2018, San Francisco, CA, USA, July 2–7, 2018, pp. 58–65. IEEE (2018). https://doi.org/10.1109/ICWS.2018.00015
49. Casalaro, G.L., Cattivera, G., Ciccozzi, F., Malavolta, I., Wortmann, A., Pelliccione, P.: Model-driven engineering for mobile robotic systems: a systematic mapping study. Softw. Syst. Model. **21**(1), 19–49 (2022). https://doi.org/10.1007/s10270-021-00908-8
50. Chauhan, U., Shah, A.: Topic modeling using latent Dirichlet allocation: a survey. ACM Comput. Surv. **54**(7), 145–114535 (2021). https://doi.org/10.1145/3462478
51. Chen, E., Qin, B., Zhu, Y., Song, W., Wang, S., Chu, C.-C.W., Yau, S.S.: SPESC-translator: towards automatically smart legal contract conversion for blockchain-based auction services. IEEE Trans. Serv. Comput. **15**(5), 3061–3076 (2022). https://doi.org/10.1109/TSC.2021.3077291
52. Chen, P.P.-S.: The entity-relationship model—toward a unified view of data. ACM Trans. Database Syst. **1**(1), 9–36 (1976). https://doi.org/10.1145/320434.320440
53. Choudhury, O., Rudolph, N., Sylla, I., Fairoza, N., Das, A.: Auto-generation of smart contracts from domain-specific ontologies and semantic rules. In: IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), iThings/GreenCom/CPSCom/SmartData 2018, Halifax, NS, Canada, July 30–August 3, 2018, pp. 963–970. IEEE (2018). https://doi.org/10.1109/Cybermatics_2018.2018.00183
54. Churchill, R., Singh, L.: The evolution of topic modeling. ACM Comput. Surv. (2021). https://doi.org/10.1145/3507900
55. Ciccio, C.D., Cecconi, A., Mendling, J., Felix, D., Haas, D., Lilek, D., Riel, F., Rumpl, A., Uhlig, P.: Blockchain-based traceability of inter-organisational business processes. In: Business Modeling and Software Design—8th International Symposium, BMSD 2018, Vienna, Austria, July 2-4, 2018, Proceedings. Lecture Notes in Business Information Processing, vol. 319, pp. 56–68, Springer (2018). https://doi.org/10.1007/978-3-319-94214-8_4
56. Clohessy, T., Acton, T., Rogers, N.: In: Treiblmaier, H., Beck, R. (eds.) Blockchain Adoption: Technological, Organisational and Environmental Considerations, pp. 47–76. Springer (2019)

57. Cohen, J.: A coefficient of agreement for nominal scales. Educ. Psychol. Meas. **20**(1), 37–46 (1960). https://doi.org/10.1177/001316446002000104
58. Conchon, S., Korneva, A., Zaïdi, F.: Verifying smart contracts with Cubicle. In: Formal Methods. FM 2019 International Workshops—Porto, Portugal, October 7–11, 2019, Revised Selected Papers, Part I. Lecture Notes in Computer Science, vol. 12232, pp. 312–324. Springer (2019). https://doi.org/10.1007/978-3-030-54994-7_23
59. Corneli, A., Naticchia, B., Spegni, F., Spalazzi, L.: Combining blockchain and BPMN coreographies for construction management. In: Proceedings of the 2021 European Conference on Computing in Construction. Computing in Construction, vol. 2, pp. 34–41. ec3 (2021a). https://doi.org/10.35490/EC3.2021.204
60. Corneli, A., Spegni, F., Bragadin, M.A., Vaccarini, M.: A smart contract-based BPMN choreography execution for management of construction processes. In: Proceedings of the 38th International Symposium on Automation and Robotics in Construction (ISARC), pp. 872–879. IAARC (2021b). https://doi.org/10.22260/ISARC2021/0118
61. Corradini, F., Marcelletti, A., Morichetta, A., Polini, A., Re, B., Tiezzi, F.: Engineering trustable choreography-based systems using blockchain. In: SAC '20: The 35th ACM/SIGAPP Symposium on Applied Computing, Online Event, [Brno, Czech Republic], March 30–April 3, 2020, pp. 1470–1479. ACM (2020). https://doi.org/10.1145/3341105.3373988
62. Corradini, F., Marcelletti, A., Morichetta, A., Polini, A., Re, B., Tiezzi, F.: ChorChain: A model-driven framework for choreography-based systems using blockchain. In: Proceedings of the 1st Italian Forum on Business Process Management Co-located with the 19th International Conference of Business Process Management (BPM 2021), Rome, Italy, September 10th, 2021. CEUR Workshop Proceedings, vol. 2952, pp. 26–32. CEUR (2021a)
63. Corradini, F., Marcelletti, A., Morichetta, A., Polini, A., Re, B., Scala, E., Tiezzi, F.: Model-driven engineering for multi-party business processes on multiple blockchains. Blockchain Res. Appl. **2**(3), 100018 (2021). https://doi.org/10.1016/j.bcra.2021.100018
64. Corradini, F., Marcelletti, A., Morichetta, A., Polini, A., Re, B., Tiezzi, F.: A choreography-driven approach for blockchain-based IoT applications. In: 2022 IEEE International Conference on Pervasive Computing and Communications Workshops and Other Affiliated Events, PerCom 2022 Workshops, Pisa, Italy, March 21–25, 2022, pp. 255–260. IEEE (2022a). https://doi.org/10.1109/PerComWorkshops53856.2022.9767513
65. Corradini, F., Marcelletti, A., Morichetta, A., Polini, A., Re, B., Tiezzi, F.: Engineering trustable and auditable choreography-based systems using blockchain. ACM Trans. Manag. Inf. Syst. **13**(3), 31–13153 (2022). https://doi.org/10.1145/3505225
66. Curty, S., Härer, F., Fill, H.-G.: Towards the comparison of blockchain-based applications using enterprise modeling. In: Proceedings of the ER Demos and Posters 2021 Co-located with 40th International Conference on Conceptual Modeling (ER 2021), St. John's, NL, Canada, October 18–21, 2021. CEUR Workshop Proceedings, vol. 2958, pp. 31–36. CEUR (2021)
67. Curty, S., Härer, F., Fill, H.-G.: Blockchain application development using model-driven engineering and low-code platforms: a survey. In: Enterprise, Business-Process and Information Systems Modeling, EMMSAD 2022, pp. 205–220. Springer (2022). https://doi.org/10.1007/978-3-031-07475-2_14
68. Curty, S., Härer, F., Fill, H.-G.: Design of Blockchain-Based Applications Using Model-Driven Engineering and Low-Code/No-Code Platforms—SLR Dataset (2023). https://doi.org/10.5281/zenodo.7839834
69. de Kruijff, J.T., Weigand, H.: Ontologies for commitment-based smart contracts. In: On the Move to Meaningful Internet Systems. OTM 2017 Conferences—Confederated International Conferences: CoopIS, C&TC, and ODBASE 2017, Rhodes, Greece, October 23–27, 2017, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10574, pp. 383–398. Springer (2017a). https://doi.org/10.1007/978-3-319-69459-7_26
70. de Kruijff, J.T., Weigand, H.: Understanding the blockchain using enterprise ontology. In: Advanced Information Systems Engineering—29th International Conference, CAiSE 2017, Essen, Germany, June 12–16, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10253, pp. 29–43. Springer (2017b). https://doi.org/10.1007/978-3-319-59536-8_3
71. de Kruijff, J.T., Weigand, H.: An introduction to commitment based smart contracts using ReactionRuleML. In: Proceedings of the 12th International Workshop on Value Modeling and Business Ontologies, VMBO 2018, Amsterdam, The Netherlands, February 26th–27th, 2018. CEUR Workshop Proceedings, vol. 2239, pp. 149–157. CEUR (2018)
72. de Kruijff, J.T., Weigand, H.: Introducing CommitRuleML for smart contracts. In: Short Paper Proceedings of the 13th International Workshop on Value Modeling and Business Ontologies, VMBO, 2019, Stockholm, Sweden, March 4–5, 2019. CEUR Workshop Proceedings, vol. 2383. CEUR (2019)
73. Dharanikota, S., Mukherjee, S., Bhardwaj, C., Rastogi, A., Lal, A.: Celestial: A smart contracts verification framework. In: Formal Methods in Computer Aided Design, FMCAD 2021, New Haven, CT, USA, October 19–22, 2021, pp. 133–142. IEEE (2021). https://doi.org/10.34727/2021/isbn.978-3-85448-046-4_22
74. Dietz, J.L.G.: Understanding and modelling business processes with DEMO. In: Akoka, J., Bouzeghoub, M., Comyn-Wattiau, I., Métais, E. (eds.) Conceptual Modeling—ER '99. Lecture Notes in Computer Science, pp. 188–202. IEEE (1999). https://doi.org/10.1007/3-540-47866-3_13
75. Dietz, J.L.G.: Enterprise Ontology: Theory and Methodology. Springer, Berlin (2006). https://doi.org/10.1007/3-540-33149-2
76. Di Ruscio, D., Kolovos, D., de Lara, J., Pierantonio, A., Tisi, M., Wimmer, M.: Low-code development and model-driven engineering: two sides of the same coin? Software and Systems Modeling (2022)
77. Dittmann, G., Sorniotti, A., Völzer, H.: Model-driven engineering for multi-party interactions on a blockchain—an example. In: Service-Oriented Computing—ICSOC 2019 Workshops—WESOACS, ASOCA, ISYCC, TBCE, and STRAPS, Toulouse, France, October 28–31, 2019, Revised Selected Papers. Lecture Notes in Computer Science, vol. 12019, pp. 181–194. Springer (2019). https://doi.org/10.1007/978-3-030-45989-5_15
78. Dwivedi, V.K., Norta, A.: A legal-relationship establishment in smart contracts: ontological semantics for programming-language development. In: Advances in Computing and Data Sciences, vol. 1440, pp. 660–676. Springer (2021). https://doi.org/10.1007/978-3-030-81462-5_58
79. Dwivedi, V.K., Norta, A.: Auto-generation of smart contracts from a domain-specific XML-based language. In: Intelligent Data Engineering and Analytics, vol. 266, pp. 549–564. Springer (2022). https://doi.org/10.1007/978-981-16-6624-7_54
80. Dwivedi, V.K., Norta, A., Wulf, A., Leiding, B., Saxena, S., Udokwu, C.: A formal specification smart-contract language for legally binding decentralized autonomous organizations. IEEE Access **9**, 76069–76082 (2021). https://doi.org/10.1109/ACCESS.2021.3081926
81. Dwivedi, V.K., Pattanaik, V., Deval, V., Dixit, A., Norta, A., Draheim, D.: Legally enforceable smart-contract languages: a systematic literature review. ACM Comput. Surv. **54**(5), 110–111034 (2021). https://doi.org/10.1145/3453475

82. Ellervee, A., Matulevičius, R., Mayer, N.: A comprehensive reference model for blockchain-based distributed ledger technology. In: Proceedings of the ER Forum 2017 and the ER 2017 Demo Track Co-located with the 36th International Conference on Conceptual Modelling (ER 2017), Valencia, Spain, November 6–9, 2017. CEUR Workshop Proceedings, vol. 1979, pp. 306–319, CEUR (2017)
83. European Law Institute (ELI): ELI Principles on Blockchain Technology, Smart Contracts and Consumer Protection (2022). https://europeanlawinstitute.eu/fileadmin/user_upload/p_eli/Publications/ELI_Principles_on_Blockchain_Technology__Smart_Contracts_and_Consumer_Protection.pdf. Last access 19 Oct. 2022
84. Evermann, J., Kim, H.M.: Workflow management on proof-of-work blockchains: implications and recommendations. SN Comput. Sci. **2**(1), 44 (2021). https://doi.org/10.1007/s42979-020-00387-6
85. Fahmideh, M., Grundy, J., Ahmad, A., Shen, J., Yan, J., Mougouei, D., Wang, P., Ghose, A., Gunawardana, A., Aickelin, U., Abedin, B.: Engineering blockchain based software systems: foundations, survey, and future directions. ACM Comput. Surv. (2022). https://doi.org/10.1145/3530813
86. Fairley, P.: Ethereum will cut back its absurd energy use. IEEE Spectr. **56**(1), 29–32 (2019)
87. Falazi, G., Hahn, M., Breitenbücher, U., Leymann, F.: Modeling and execution of blockchain-aware business processes. SICS Softw. Intensive Cyber Phys. Syst. **34**(2–3), 105–116 (2019). https://doi.org/10.1007/s00450-019-00399-5
88. Falazi, G., Hahn, M., Breitenbücher, U., Leymann, F., Yussupov, V.: Process-based composition of permissioned and permissionless blockchain smart contracts. In: 23rd IEEE International Enterprise Distributed Object Computing Conference, EDOC 2019, Paris, France, October 28–31, 2019, pp. 77–87. IEEE (2019b). https://doi.org/10.1109/EDOC.2019.00019
89. Fayyad, U.: Knowledge discovery in databases: an overview. In: Džeroski, S., Lavrač, N. (eds.) Relational Data Mining, pp. 28–47. Springer, Berlin (2001). https://doi.org/10.1007/978-3-662-04599-2_2
90. Ferstl, O.K., Sinz, E.J.: Modeling of business systems using SOM. In: Bernus, P., Mertins, K., Schmidt, G. (eds.) Handbook on Architectures of Information Systems. International Handbooks on Information Systems, pp. 347–367. Springer, Berlin (2006). https://doi.org/10.1007/3-540-26661-5_15
91. Fettke, P., Loos, P.: Perspectives on reference modeling. In: Reference Modeling for Business Systems Analysis, pp. 1–21. IGI Global (2007). https://doi.org/10.4018/978-1-59904-054-7.ch001
92. Fill, H.-G., Meier, A. (eds.): Blockchain - Grundlagen, Anwendungsszenarien und Nutzungspotenziale. Edition HMD. Springer, Berlin (2020). https://doi.org/10.1007/978-3-658-28006-2
93. Fill, H.-G., Meier, A.: Blockchain Kompakt. Springer, Berlin (2020). https://doi.org/10.1007/978-3-658-27461-0
94. Fill, H.-G., Fettke, P., Rinderle-Ma, S.: Catchword: blockchains and enterprise modeling. Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model. **15**, 16–1168 (2020). https://doi.org/10.18417/emisa.15.16
95. Fill, H.-G., Härer, F., Muff, F., Curty, S.: Towards augmented enterprise models as low-code interfaces to digital systems. In: International Symposium on Business Modeling and Software Design, pp. 343–352. Springer (2021)
96. Fox, M.S.: The TOVE project towards a common-sense model of the enterprise. In: Industrial and Engineering Applications of Artificial Intelligence and Expert Systems. Lecture Notes in Computer Science, pp. 25–34. Springer (1992). https://doi.org/10.1007/BFb0024952
97. Fox, M.S., Grüninger, M.: Enterprise modeling. AI Mag. **19**(3), 109–121 (1998). https://doi.org/10.1609/aimag.v19i3.1399
98. Frantz, C., Nowostawski, M.: From institutions to code: towards automated generation of smart contracts. In: 2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS*W), Augsburg, Germany, September 12–16, 2016, pp. 210–215. IEEE (2016). https://doi.org/10.1109/FAS-W.2016.53
99. Franz, F., Fertig, T., Schütz, A.E.: Democratization of smart contracts: a prototype for automated contract generation. In: IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2020, Toronto, ON, Canada, May 2–6, 2020, pp. 1–3. IEEE (2020). https://doi.org/10.1109/ICBC48266.2020.9169479
100. Fraternali, P., Gonzalez, S.L.H., Frigerio, M., Righetti, M.: Model-driven development of distributed ledger applications. In: Database Systems for Advanced Applications. DASFAA 2022 International Workshops, vol. 13248, pp. 104–119. Springer (2022). https://doi.org/10.1007/978-3-031-11217-1_8
101. Garamvölgyi, P., Kocsis, I., Gehl, B., Klenik, A.: Towards model-driven engineering of smart contracts for cyber-physical systems. In: 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops, DSN Workshops 2018, Luxembourg, June 25–28, 2018, pp. 134–139. IEEE (2018). https://doi.org/10.1109/DSN-W.2018.00052
102. García-Bañuelos, L., Ponomarev, A., Dumas, M., Weber, I.: Optimized execution of business processes on blockchain. In: Business Process Management—15th International Conference, BPM 2017, Barcelona, Spain, September 10-15, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10445, pp. 130–146. Springer (2017). https://doi.org/10.1007/978-3-319-65000-5_8
103. García-García, J.A., Sánchez-Gómez, N., Lizcano, D., Escalona Cuaresma, M.J., Wojdynski, T.: Using blockchain to improve collaborative business process management: systematic literature review. IEEE Access **8**, 142312–142336 (2020). https://doi.org/10.1109/ACCESS.2020.3013911
104. Gogolla, M., Goos, G., Hartmanis, J. (eds.): An Extended Entity-Relationship Model. Lecture Notes in Computer Science, vol. 767. Springer (1994). https://doi.org/10.1007/3-540-57648-7
105. Gómez, C., Pérez Blanco, F.J., Vara, J.M., De Castro, V., Marcos, E.: Design and development of Smart Contracts for E-government through Value and Business Process Modeling. In: Hawaii International Conference on System Sciences. HICSS (2021). https://doi.org/10.24251/HICSS.2021.254
106. Gonczol, P., Katsikouli, P., Herskind, L., Dragoni, N.: Blockchain implementations and use cases for supply chains—A survey. IEEE Access **8**, 11856–11871 (2020). https://doi.org/10.1109/ACCESS.2020.2964880
107. Gordijn, J., Akkermans, H.: Designing and evaluating e-business models. IEEE Intell. Syst. **16**(4), 11–17 (2001). https://doi.org/10.1109/5254.941353
108. Grishchenko, I., Maffei, M., Schneidewind, C.: Foundations and tools for the static analysis of Ethereum smart contracts. In: Computer Aided Verification—30th International Conference, CAV 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14–17, 2018, Proceedings, Part I. Lecture Notes in Computer Science, vol. 10981, pp. 51–78. Springer (2018). https://doi.org/10.1007/978-3-319-96145-3_4
109. Górski, T.: Reconfigurable smart contracts for renewable energy exchange with re-use of verification rules. Appl. Sci. (2022). https://doi.org/10.3390/app12115339
110. Górski, T., Bednarski, J.: Modeling of smart contracts in blockchain solution for renewable energy grid. In: Computer Aided Systems Theory—EUROCAST 2019—17th International Conference, Las Palmas de Gran Canaria, Spain, February 17-22, 2019, Revised Selected Papers, Part I. Lecture Notes in Com-

puter Science, vol. 12013, pp. 507–514. Springer (2019). https://doi.org/10.1007/978-3-030-45093-9_61
111. Górski, T., Bednarski, J.: Applying model-driven engineering to distributed ledger deployment. IEEE Access **8**, 118245–118261 (2020). https://doi.org/10.1109/ACCESS.2020.3005519
112. Górski, T., Bednarski, J.: Modeling of distributed ledger deployment view. Int. J. Electron. Telecommun. **66**(4), 619–625 (2020). https://doi.org/10.24425/ijet.2020.134020
113. Górski, T., Bednarski, J.: Transformation of the UML deployment model into a distributed ledger network configuration. In: 15th IEEE International Conference of System of Systems Engineering, SoSE 2020, Budapest, Hungary, June 2-4, 2020, pp. 255–260. IEEE (2020). https://doi.org/10.1109/SoSE50414.2020.9130492
114. Guerreiro, S., Silva, D., Rosado, T., Vasconcelos, A., Correia, M., Sousa, P.: Decentralized business process control using blockchain - An experience report from two applications: food supply chain and car registration. Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model. **15**, 15–11541 (2020). https://doi.org/10.18417/emisa.15.15
115. Haarmann, S.: Executing DMN decisions on the blockchain. In: Blockchain and Robotic Process Automation, pp. 43–53. Springer (2021). https://doi.org/10.1007/978-3-030-81409-0_4
116. Haarmann, S., Batoulis, K., Nikaj, A., Weske, M.: DMN decision execution on the Ethereum blockchain. In: Advanced Information Systems Engineering—30th International Conference, CAiSE 2018, Tallinn, Estonia, June 11–15, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10816, pp. 327–341. Springer (2018). https://doi.org/10.1007/978-3-319-91563-0_20
117. Haarmann, S., Batoulis, K., Nikaj, A., Weske, M.: Executing collaborative decisions confidentially on blockchains. In: Business Process Management: Blockchain and Central and Eastern Europe Forum—BPM 2019 Blockchain and CEE Forum, Vienna, Austria, September 1–6, 2019, Proceedings. Lecture Notes in Business Information Processing, vol. 361, pp. 119–135. Springer (2019). https://doi.org/10.1007/978-3-030-30429-4_9
118. Hamadi, Y.B., Heng, S., Wautelet, Y.: Using i*-based organizational modeling to support blockchain-oriented software engineering: case study in supply chain management. In: Research and Innovation Forum 2020—Disruptive Technologies in Times of Change, RIIFORUM 2020, Athens, Greece, 15–17 April 2020, pp. 495–515. Springer (2020). https://doi.org/10.1007/978-3-030-62066-0_38
119. Hamdaqa, M., Met, L.A.P., Qasse, I.A.: iContractML 2.0: a domain-specific language for modeling and deploying smart contracts onto multiple blockchain platforms. Inf. Softw. Technol. **144**, 106762 (2022). https://doi.org/10.1016/j.infsof.2021.106762
120. Hamdaqa, M., Metz, L.A.P., Qasse, I.A.: iContractML: a domain-specific language for modeling and deploying smart contracts onto multiple blockchain platforms. In: SAM '20: 12th System Analysis and Modelling Conference, Virtual Event, Canada, October 19–20, 2020, pp. 34–43. ACM (2020). https://doi.org/10.1145/3419804.3421454
121. Härer, F.: Process modeling in decentralized organizations utilizing blockchain consensus. Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model. **15**, 13–11317 (2020). https://doi.org/10.18417/emisa.15.13
122. Härer, F., Fill, H.-G.: A comparison of approaches for visualizing blockchains and smart contracts. In: 22nd International Legal Informatics Symposium/22. Internationales Rechtsinformatik Symposion (IRIS 2019), pp. 527–537. Editions Weblaw (2019)
123. Härer, F., Fill, H.-G.: Past trends and future prospects in conceptual modeling—a bibliometric analysis. In: Dobbie, G., Frank, U., Kappel, G., Liddle, S.W., Mayr, H.C. (eds.) 39th International Conference on Conceptual Modeling (ER 2020). Springer (2020)
124. Hayes, P.J., Patel-Schneider, P.F.: RDF 1.1 Semantics (2014). https://www.w3.org/TR/rdf11-mt/. Last access 19 Oct. 2022
125. He, X., Qin, B., Zhu, Y., Chen, X., Liu, Y.: SPESC: A specification language for smart contracts. In: 2018 IEEE 42nd Annual Computer Software and Applications Conference, COMPSAC 2018, Tokyo, Japan, 23–27 July 2018, Volume 1, pp. 132–137. IEEE (2018). https://doi.org/10.1109/COMPSAC.2018.00025
126. He, X.: Modeling and analyzing smart contracts using predicate transition nets. In: 20th IEEE International Conference on Software Quality, Reliability and Security Companion, QRS Companion 2020, Macau, China, December 11–14, 2020, pp. 108–115. IEEE (2020). https://doi.org/10.1109/QRS-C51114.2020.00029
127. Hearn, M.: Corda: a distributed ledger. Whitepaper (2021). https://www.corda.net/wp-content/uploads/2021/11/corda-technical-whitepaper.pdf
128. Heckel, R., Erum, Z., Rahmi, N., Pul, A.: Visual smart contracts for DAML. In: Graph Transformation, vol. 13349, pp. 137–154. Springer (2022). https://doi.org/10.1007/978-3-031-09843-7_8
129. Hector, U.-R., Boris, C.-L.: BLONDiE: blockchain ontology with dynamic extensibility. CoRR (2020). https://doi.org/10.48550/ARXIV.2008.09518
130. Henry, T., Brahem, A., Laga, N., Hatin, J., Gaaloul, W., Benatallah, B.: Trustworthy cross-organizational collaborations with hybrid on/off-chain declarative choreographies. In: Service-Oriented Computing—19th International Conference, ICSOC 2021, Virtual Event, November 22–25, 2021, Proceedings. Lecture Notes in Computer Science, vol. 13121, pp. 81–96. Springer (2021). https://doi.org/10.1007/978-3-030-91431-8_6
131. Holotiuk, F., Moormann, J.: Organizational adoption of digital innovation: The case of blockchain technology. In: 26th European Conference on Information Systems (ECIS 2018). AIS (2018)
132. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M.: SWRL: a Semantic Web Rule Language Combining OWL and RuleML (2004). https://www.w3.org/Submission/SWRL/. Last access 19 Oct. 2022
133. Hu, B., Zhang, Z., Liu, J., Liu, Y., Yin, J., Lu, R., Lin, X.: A comprehensive survey on smart contract construction and execution: paradigms, tools, and systems. Patterns **2**(2), 100179 (2021). https://doi.org/10.1016/j.patter.2020.100179
134. Hyperledger Foundation: An Introduction to Hyperledger (2018). https://www.hyperledger.org/wp-content/uploads/2018/08/HL_Whitepaper_IntroductiontoHyperledger.pdf. Last access 19 Oct. 2022
135. Jiang, S., Ræder, T.B.: Experience on using ArchiMate models for modelling blockchain-enhanced value chains. In: EASE 2022: The International Conference on Evaluation and Assessment in Software Engineering 2022, Gothenburg, Sweden, June 13–15, 2022, pp. 375–382. ACM (2022). https://doi.org/10.1145/3530019.3531346
136. Jovanovic, M., Kostić, N., Sebastian, I.M., Sedej, T.: Managing a blockchain-based platform ecosystem for industry-wide adoption: the case of TradeLens. Technol. Forecast. Soc. Change **184**, 121981 (2022). https://doi.org/10.1016/j.techfore.2022.121981
137. Jurgelaitis, M., Ceponiene, L., Butkiene, R.: Solidity code generation from UML state machines in model-driven smart contract development. IEEE Access **10**, 33465–33481 (2022). https://doi.org/10.1109/ACCESS.2022.3162227
138. Jurgelaitis, M., Drungilas, V., Ceponiene, L., Butkiene, R., Vaiciukynas, E.: Modelling principles for blockchain-based implementation of business or scientific processes. In: Proceedings of the International Conference on Information Technologies, IVUS 2019, Kaunas, Lithuania, April 25, 2019. CEUR Workshop Proceedings, vol. 2470, pp. 43–47. CEUR (2019)
139. Jurgelaitis, M., Drungilas, V., Ceponiene, L., Vaiciukynas, E., Butkiene, R., Ceponis, J.: Smart contract code generation from

platform specific model for Hyperledger Go. In: Trends and Applications in Information Systems and Technologies—Volume 4, WorldCIST 2021, Terceira Island, Azores, Portugal, 30 March–2 April, 2021. Advances in Intelligent Systems and Computing, vol. 1368, pp. 63–73. Springer (2021). https://doi.org/10.1007/978-3-030-72654-6_7
140. Kherbouche, M., Pisoni, G., Molnár, B.: Model to program and blockchain approaches for business processes and workflows in finance. Appl. Syst. Innov. **5**(1), 10 (2022). https://doi.org/10.3390/asi5010010
141. Kim, H.M., Laskowski, M.: Toward an ontology-driven blockchain design for supply-chain provenance. Intell. Syst. Account. Finance Manag. **25**(1), 18–27 (2018). https://doi.org/10.1002/isaf.1424
142. Klinger, P., Bodendorf, F.: Blockchain-based cross-organizational execution framework for dynamic integration of process collaborations. In: Entwicklungen, Chancen und Herausforderungen der Digitalisierung: Proceedings der 15. Internationalen Tagung Wirtschaftsinformatik, WI 2020, Potsdam, Germany, March 9–11, 2020. Zentrale Tracks, pp. 893–908. GITO (2020). https://doi.org/10.30844/wi_2020_i2-klinger
143. Kolb, J., Yang, J., Katz, R.H., Culler, D.E.: Quartz: A framework for engineering secure smart contracts. Technical Report UCB/EECS-2020-178, EECS Department, University of California, Berkeley (2020)
144. Ladleif, J., Weske, M.: A unifying model of legal smart contracts. In: Conceptual Modeling—38th International Conference, ER 2019, Salvador, Brazil, November 4–7, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11788, pp. 323–337. Springer (2019). https://doi.org/10.1007/978-3-030-33223-5_27
145. Ladleif, J., Friedow, C., Weske, M.: An architecture for multi-chain business process choreographies. In: Business Information Systems—23rd International Conference, BIS 2020, Colorado Springs, CO, USA, June 8–10, 2020, Proceedings. Lecture Notes in Business Information Processing, vol. 389, pp. 184–196. Springer (2020). https://doi.org/10.1007/978-3-030-53337-3_14
146. Ladleif, J., Weske, M., Weber, I.: Modeling and enforcing blockchain-based choreographies. In: Business Process Management—17th International Conference, BPM 2019, Vienna, Austria, September 1–6, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11675, pp. 69–85. Springer (2019). https://doi.org/10.1007/978-3-030-26619-6_7
147. Lallai, G., Pinna, A., Marchesi, M., Tonelli, R.: Software engineering for DApp smart contracts managing workers contracts. In: Proceedings of the 3rd Distributed Ledger Technology Workshop Co-located with ITASEC 2020, Ancona, Italy, February 4, 2020. CEUR Workshop Proceedings, vol. 2580, CEUR (2020)
148. Lamela Seijas, P., Thompson, S.: Marlowe: financial contracts on blockchain. In: Leveraging Applications of Formal Methods, Verification and Validation. Industrial Practice—8th International Symposium, ISoLA 2018, Limassol, Cyprus, November 5–9, 2018, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 11247, pp. 356–375. Springer (2018). https://doi.org/10.1007/978-3-030-03427-6_27
149. Lamela Seijas, P., Nemish, A., Smith, D., Thompson, S.: Marlowe: Implementing and analysing financial contracts on blockchain. In: Financial Cryptography and Data Security—FC 2020 International Workshops, AsiaUSEC, CoDeFi, VOTING, and WTSC, Kota Kinabalu, Malaysia, February 14, 2020, Revised Selected Papers. Lecture Notes in Computer Science, vol. 12063, pp. 496–511. Springer (2020). https://doi.org/10.1007/978-3-030-54455-3_35
150. Landis, J.R., Koch, G.G.: The measurement of observer agreement for categorical data. Biometrics **33**(1), 159–174 (1977). https://doi.org/10.2307/2529310
151. Lankhorst, M.M., Proper, H.A., Jonkers, H.: The architecture of the ArchiMate language. In: Enterprise, Business-Process and Information Systems Modeling. Lecture Notes in Business Information Processing, pp. 367–380. Springer (2009). https://doi.org/10.1007/978-3-642-01862-6_30
152. Levasseur, O., Iqbal, M., Matulevičius, R.: Survey of model-driven engineering techniques for blockchain-based applications. In: Proceedings of the Forum at Practice of Enterprise Modeling 2021 (PoEM-Forum 2021) (PoEM 2021), Riga, Latvia, November 24–26, 2021. CEUR Workshop Proceedings, vol. 3045, pp. 11–20. CEUR (2021)
153. LiBin, M.T., WaiShiang, C., Khairuddin, M.A.B., Mit, E., Erianda, A.: Agent-oriented modelling for blockchain application development: feasibility study. JOIV Int. J. Inform. Vis. **5**(3), 248–255 (2021). https://doi.org/10.30630/joiv.5.3.670
154. Liu, C., Bodorik, P., Jutla, D.: From BPMN to smart contracts on blockchains: transforming BPMN to DE-HSM multi-modal model. In: 2021 International Conference on Engineering and Emerging Technologies (ICEET), pp. 1–7. IEEE (2021). https://doi.org/10.1109/ICEET53442.2021.9659771
155. Liu, C., Bodorik, P., Jutla, D.: Automating smart contract generation on blockchains using multi-modal modeling. J. Adv. Inf. Technol. (2022). https://doi.org/10.12720/jait.13.3.213-223
156. Liu, C., Bodorik, P., Jutla, D.N.: A tool for moving blockchain computations off-chain. In: BSCI '21: Proceedings of the 3rd ACM International Symposium on Blockchain and Secure Critical Infrastructure, Virtual Event, Hong Kong, June 7, 2021, pp. 103–109. ACM (2021). https://doi.org/10.1145/3457337.3457848
157. Liu, J., Liu, Z.: A survey on security verification of blockchain smart contracts. IEEE Access **7**, 77894–77904 (2019). https://doi.org/10.1109/ACCESS.2019.2921624
158. Liu, Y., Li, Y., Lin, S., Yan, Q.: ModCon: a model-based testing platform for smart contracts. In: ESEC/FSE '20: 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Virtual Event, USA, November 8–13, 2020, pp. 1601–1605. ACM (2020). DOIurl-https://doi.org/10.1145/3368089.3417939
159. López-Pintado, O., García-Bañuelos, L., Dumas, M., Weber, I.: Caterpillar: a blockchain-based business process management system. In: Proceedings of the BPM Demo Track and BPM Dissertation Award Co-located with 15th International Conference on Business Process Modeling (BPM 2017), Barcelona, Spain, September 13, 2017. CEUR Workshop Proceedings, vol. 1920, CEUR (2017)
160. López-Pintado, O., García-Bañuelos, L., Dumas, M., Weber, I., Ponomarev, A.: Caterpillar: a business process execution engine on the Ethereum blockchain. Softw. Pract. Exp. **49**(7), 1162–1193 (2019). https://doi.org/10.1002/spe.2702
161. López-Pintado, O., Dumas, M., García-Bañuelos, L., Weber, I.: Dynamic role binding in blockchain-based collaborative business processes. In: Advanced Information Systems Engineering - 31st International Conference, CAiSE 2019, Rome, Italy, June 3–7, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11483, pp. 399–414. Springer (2019b). https://doi.org/10.1007/978-3-030-21290-2_25
162. López-Pintado, O., Dumas, M., García-Bañuelos, L., Weber, I.: Interpreted execution of business process models on blockchain. In: 23rd IEEE International Enterprise Distributed Object Computing Conference, EDOC 2019, Paris, France, October 28–31, 2019, pp. 206–215. IEEE (2019c). https://doi.org/10.1109/EDOC.2019.00033
163. López-Pintado, O., Dumas, M., García-Bañuelos, L., Weber, I.: Controlled flexibility in blockchain-based collaborative business processes. Inf. Syst. **104**, 101622 (2022). https://doi.org/10.1016/j.is.2020.101622

164. Loukil, F., Boukadi, K., Abed, M., Guegan, C.G.: Decentralized collaborative business process execution using blockchain. World Wide Web **24**(5), 1645–1663 (2021). https://doi.org/10.1007/s11280-021-00901-7
165. Lu, Q., Tran, A.B., Weber, I., O'Connor, H., Rimba, P., Xu, X., Staples, M., Zhu, L., Jeffery, R.: Integrated model-driven engineering of blockchain applications for business processes and asset management. Softw. Pract. Exp. **51**(5), 1059–1079 (2021). https://doi.org/10.1002/spe.2931
166. Madsen, M.F., Gaub, M., Høgnason, T., Kirkbro, M.E., Slaats, T., Debois, S.: Collaboration among adversaries: distributed workflow execution on a blockchain. In: Symposium on Foundations and Applications of Blockchain, p. 8. ITU (2018)
167. Mao, D., Wang, F., Wang, Y., Hao, Z.: Visual and user-defined smart contract designing system based on automatic coding. IEEE Access **7**, 73131–73143 (2019). https://doi.org/10.1109/ACCESS.2019.2920776
168. Marchesi, L., Marchesi, M., Tonelli, R.: ABCDE–agile block chain DApp engineering. Blockchain Res. Appl. **1**(1–2), 100002 (2020). https://doi.org/10.1016/j.bcra.2020.100002
169. Marchesi, L., Mannaro, K., Marchesi, M., Tonelli, R.: Automatic generation of Ethereum-based smart contracts for agri-food traceability system. IEEE Access **10**, 50363–50383 (2022). https://doi.org/10.1109/ACCESS.2022.3171045
170. Marchesi, M., Marchesi, L., Tonelli, R.: An agile software engineering method to design blockchain applications. In: Proceedings of the 14th Central and Eastern European Software Engineering Conference Russia on ZZZ—CEE-SECR '18, pp. 1–8. ACM (2018). https://doi.org/10.1145/3290621.3290627
171. Martin, C.F.: Second-generation CASE tools: a challenge to vendors. IEEE Softw. **5**(2), 46–49 (1988). https://doi.org/10.1109/52.2010
172. Mavridou, A., Laszka, A.: Designing secure Ethereum smart contracts: a finite state machine based approach. In: Financial Cryptography and Data Security—22nd International Conference, FC 2018, Nieuwpoort, Curaçao, February 26–March 2, 2018, Revised Selected Papers. Lecture Notes in Computer Science, vol. 10957, pp. 523–540, Springer (2018a). https://doi.org/10.1007/978-3-662-58387-6_28
173. Mavridou, A., Laszka, A.: Tool demonstration: FSolidM for designing secure Ethereum smart contracts. In: Principles of Security and Trust—7th International Conference, POST 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14–20, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10804, pp. 270–277. Springer (2018b). https://doi.org/10.1007/978-3-319-89722-6_11
174. Mavridou, A., Laszka, A., Stachtiari, E., Dubey, A.: VeriSolid: Correct-by-design smart contracts for Ethereum. In: Financial Cryptography and Data Security—23rd International Conference, FC 2019, Frigate Bay, St. Kitts and Nevis, February 18-22, 2019, Revised Selected Papers. Lecture Notes in Computer Science, vol. 11598, pp. 446–465. Springer (2019). https://doi.org/10.1007/978-3-030-32101-7_27
175. McCarthy, W.E.: The REA accounting model: a generalized framework for accounting systems in a shared data environment. Account. Rev. **57**(3), 554–578 (1982)
176. McHugh, M.L.: Interrater reliability: the kappa statistic. Biochem. Med. **22**(3), 276–282 (2012)
177. Meng, B., Li, M., Beckmann, B., Nishida, Y., Carbone, J., Yang, D., Durling, M.: Towards developing trusted smart contracts in Simulink. In: Proceedings of the Workshops Co-organized with the 13th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modelling (PoEM 2020), On-line (originally Located in Riga, Latvia), November 26, 2020. CEUR Workshop Proceedings, vol. 2749, pp. 35–46. CEUR (2020)
178. Merlec, M.M., Lee, Y.K., In, H.P.: SmartBuilder: A block-based visual programming framework for smart contract development. In: 2021 IEEE International Conference on Blockchain, Blockchain 2021, Melbourne, Australia, December 6–8, 2021, pp. 90–94. IEEE (2021). https://doi.org/10.1109/Blockchain53845.2021.00023
179. Milani, F., García-Bañuelos, L., Filipova, S., Markovska, M.: Modelling blockchain-based business processes: a comparative analysis of BPMN vs CMMN. Bus. Process. Manag. J. **27**(2), 638–657 (2021). https://doi.org/10.1108/BPMJ-06-2020-0263
180. Mirković, A., Terzić, B., Gajić, D., Nenić, M., Luković, I.: A model-driven approach to establishment of private blockchain business networks. In: Proceedings of the 9th International Conference on Information Society and Technology. ISOS Conference Proceedings Series, pp. 10–14. ISOS (2019)
181. Morales-Sandoval, M., Molina, J.A., Castro, H.M.M., González-Compeán, J.L.: Blockchain support for execution, monitoring and discovery of inter-organizational business processes. PeerJ Comput. Sci. **7**, 731 (2021). https://doi.org/10.7717/peerj-cs.731
182. Muff, F., Härer, F., Fill, H.-G.: Trends in academic and industrial research on business process management—a computational literature analysis. In: 55th Hawaii International Conference on System Sciences (HICSS-55). HICSS (2022)
183. Murray, Y., Anisi, D.A.: Survey of formal verification methods for smart contracts on blockchain. In: 10th IFIP International Conference on New Technologies, Mobility and Security, NTMS 2019, Canary Islands, Spain, June 24–26, 2019, pp. 1–6. IEEE (2019). https://doi.org/10.1109/NTMS.2019.8763832
184. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system. Whitepaper (2008). https://bitcoin.org/bitcoin.pdf
185. Nakamura, H., Miyamoto, K., Kudo, M.: Inter-organizational business processes managed by blockchain. In: Web Information Systems Engineering—WISE 2018—19th International Conference, Dubai, United Arab Emirates, November 12–15, 2018, Proceedings, Part I. Lecture Notes in Computer Science, vol. 11233, pp. 3–17. Springer (2018). https://doi.org/10.1007/978-3-030-02922-7_1
186. Nelaturu, K., Mavridou, A., Veneris, A.G., Laszka, A.: Verified development and deployment of multiple interacting smart contracts with VeriSolid. In: IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2020, Toronto, ON, Canada, May 2–6, 2020, pp. 1–9. IEEE (2020). https://doi.org/10.1109/ICBC48266.2020.9169428
187. Nelaturu, K., Mavridou, A., Stachtiari, E., Veneris, A., Laszka, A.: Correct-by-design interacting smart contracts and a systematic approach for verifying ERC20 and ERC721 contracts with VeriSolid. IEEE Trans. Dependable Secur. Comput. (2022). https://doi.org/10.1109/TDSC.2022.3200840
188. Newman, D., Asuncion, A., Smyth, P., Welling, M.: Distributed algorithms for topic models. J. Mach. Learn. Res. **10**, 1801–1828 (2009)
189. Nguyen, C.T., Hoang, D.T., Nguyen, D.N., Niyato, D., Nguyen, H.T., Dutkiewicz, E.: Proof-of-stake consensus mechanisms for future blockchain networks: fundamentals, applications and opportunities. IEEE Access **7**, 85727–85745 (2019)
190. Nissl, M., Sallinger, E.: Towards bridging traditional and smart contracts with Datalog-based languages. In: Proceedings of the 4th International Workshop on the Resurgence of Datalog in Academia and Industry (Datalog-2.0 2022) Co-located with the 16th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR 2022), Genova-Nervi, Italy, September 5, 2022. CEUR Workshop Proceedings, vol. 3203, pp. 68–82. CEUR (2022)
191. Norcini, J.J.: Standards and reliability in evaluation: when rules of thumb don't apply. Acad. Med. J. Assoc. Am. Med. Coll.

74(10), 1088–1090 (1999). https://doi.org/10.1097/00001888-199910000-00010
192. Nousias, N., Tsakalidis, G., Petridou, S.G., Vergidis, K.: Modelling the development and deployment of decentralized applications in Ethereum blockchain: a BPMN-based approach. In: Decision Support Systems XII: Decision Support Addressing Modern Industry, Business, and Societal Needs - 8th International Conference on Decision Support System Technology, ICDSST 2022, Thessaloniki, Greece, May 23–25, 2022, Proceedings. Lecture Notes in Business Information Processing, vol. 447, pp. 55–67. Springer (2022). https://doi.org/10.1007/978-3-031-06530-9_5
193. OASIS Web Services Business Process Execution Language (WSBPEL) TC: Business Process Execution Language (2007). http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html. Last access 19 Oct. 2022
194. Olivé, A.: The conceptual schema of Ethereum. In: Conceptual Modeling—39th International Conference, ER 2020, Vienna, Austria, November 3–6, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12400, pp. 418–428. Springer (2020). https://doi.org/10.1007/978-3-030-62522-1_31
195. Pande, C., Fill, H.-G., Hinkelmann, K.: A computational literature analysis of conversational AI research with a focus on the coaching domain. In: Proceedings of the Society 5.0 Conference 2022. EPIC Series in Computing (2022). https://doi.org/10.29007/lh9r
196. Panduwinata, F., Yugopuspito, P.: BPMN approach in blockchain with hyperledger composer and smart contract: reservation-based parking system. In: 2019 5th International Conference on New Media Studies (CONMEDIA), pp. 89–93. IEEE (2019). https://doi.org/10.1109/CONMEDIA46929.2019.8981845
197. Park, W., Lee, H., Choi, J.: Formal modeling of smart contract-based trading system. In: 23rd International Conference on Advanced Communication Technology, ICACT 2021, Pyeongchang, South Korea, February 7–10, 2021, pp. 48–52. IEEE (2021). https://doi.org/10.23919/ICACT51234.2021.9370462
198. Perrelet, S., Fill, H.-G., Dibbern, J.: A modeling approach for blockchain-inspired business models: an extension of the E3-Value method. In: Hawaii International Conference on System Sciences. HICSS (2022). https://doi.org/10.24251/HICSS.2022.558
199. Petri, C.A.: Kommunikation mit Automaten. PhD thesis, Fakultät für Mathematik und Physik, Technische Hochschule, Darmstadt (1962)
200. Petri, C.A., Reisig, W.: Petri Net. Scholarpedia **3**(4), 6477 (2008). https://doi.org/10.4249/scholarpedia.6477
201. Petrović, N., Tošić, M.: Semantic approach to smart contract verification. Facta Univ. Ser. Autom. Control Robot. **19**(1), 021–037 (2020). https://doi.org/10.22190/FUACR2001021P
202. Pinna, A., Tonelli, R.: On the use of Petri Nets in smart contracts modeling, generation and verification. In: IEEE International Conference on Software Analysis, Evolution and Reengineering, SANER 2022, Honolulu, HI, USA, March 15–18, 2022, pp. 1207–1211. IEEE (2022). https://doi.org/10.1109/SANER53432.2022.00142
203. Poels, G., Kaya, F., Verdonck, M., Gordijn, J.: Early identification of potential distributed ledger technology business cases using e3value models. In: Advances in Conceptual Modeling—ER 2019 Workshops FAIR, MREBA, EmpER, MoBiD, OntoCom, and ER Doctoral Symposium Papers, Salvador, Brazil, November 4-7, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11787, pp. 70–80. Springer (2019). https://doi.org/10.1007/978-3-030-34146-6_7
204. Purnell, K., Schwitter, R.: User-defined smart contracts using answer set programming. In: AI 2021: Advances in Artificial Intelligence—34th Australasian Joint Conference, AI 2021, Sydney, NSW, Australia, February 2–4, 2022, Proceedings. Lecture Notes in Computer Science, vol. 13151, pp. 291–303. Springer (2022). https://doi.org/10.1007/978-3-030-97546-3_24
205. Qasse, I.A., Mishra, S., Hamdaqa, M.: Chat2Code: towards conversational concrete syntax for model specification and code generation, the case of smart contracts. CoRR (2021). https://doi.org/10.48550/ARXIV.2112.11101
206. Qasse, I.A., Mishra, S., Hamdaqa, M.: iContractBot: a chatbot for smart contracts' specification and code generation. In: 3rd IEEE/ACM International Workshop on Bots in Software Engineering, BotSE@ICSE 2021, Madrid, Spain, June 4, 2021, pp. 35–38. IEEE (2021b). https://doi.org/10.1109/BotSE52550.2021.00015
207. Regnath, E., Steinhorst, S.: SmaCoNat: Smart contracts in natural language. In: 2018 Forum on Specification and Design Languages, FDL 2018, Garching, Germany, September 10–12, 2018, pp. 5–16. IEEE (2018). https://doi.org/10.1109/FDL.2018.8524068
208. Rocha, H., Ducasse, S.: Preliminary steps towards modeling blockchain oriented software. In: 1st IEEE/ACM International Workshop on Emerging Trends in Software Engineering for Blockchain, WETSEB@ICSE 2018, Gothenburg, Sweden, May 27–June 3, 2018, pp. 52–57. ACM (2018). https://doi.org/10.1145/3194113.3194123
209. Rokis, K., Kirikova, M.: Challenges of low-code/no-code software development: A literature review. In: Nazaruka, Ē., Sandkuhl, K., Seigerroth, U. (eds.) Perspectives in Business Informatics Research. Lecture Notes in Business Information Processing, pp. 3–17. Springer (2022). https://doi.org/10.1007/978-3-031-16947-2_1
210. Rosa-Bilbao, J., Boubeta-Puig, J., Rutle, A.: EDALoCo: enhancing the accessibility of blockchains through a low-code approach to the development of event-driven applications for smart contract management. Comput. Stand. Interfaces **84**, 103676 (2023). https://doi.org/10.1016/j.csi.2022.103676
211. Roussille, H., Gürcan, Ö., Michel, F.: AGR4BS: a generic multi-agent organizational model for blockchain systems. Big Data Cogn. Comput. **6**(1), 1 (2022). https://doi.org/10.3390/bdcc6010001
212. Rule Markup Initiative: RuleML—W3C RIF-WG Wiki (2005). https://www.w3.org/2005/rules/wg/wiki/RuleML. Last access 19 Oct. 2022
213. Sahay, A., Indamutsa, A., Di Ruscio, D., Pierantonio, A.: Supporting the understanding and comparison of low-code development platforms. In: SEAA Conference, pp. 171–178. IEEE (2020)
214. Sánchez-Gómez, N., Morales-Trujillo, L., Valderrama, J.T.: Towards an approach for applying early testing to smart contracts. In: Proceedings of the 15th International Conference on Web Information Systems and Technologies, WEBIST 2019, Vienna, Austria, September 18–20, 2019, pp. 445–453. ScitePress (2019). https://doi.org/10.5220/0008386004450453
215. Sánchez-Gómez, N., Torres-Valderrama, J., García-García, J.A., Gutiérrez, J.J., Escalona Cuaresma, M.J.: Model-based software design and testing in blockchain smart contracts: a systematic literature review. IEEE Access **8**, 164556–164569 (2020). https://doi.org/10.1109/ACCESS.2020.3021502
216. Sato, N., Tateishi, T., Amano, S.: Formal requirement enforcement on smart contracts based on linear dynamic logic. In: IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), iThings/GreenCom/CPSCom/SmartData 2018, Halifax, NS, Canada, July 30–August 3, 2018, pp. 945–954. IEEE (2018). https://doi.org/10.1109/Cybermatics_2018.2018.00181

217. Schindelmann, M., Klinger, P., Bodendorf, F.: A subscription service for automated communication and fair cost distribution in collaborative blockchain-based business processes. In: Entwicklungen, Chancen und Herausforderungen der Digitalisierung: Proceedings der 15. Internationalen Tagung Wirtschaftsinformatik, WI 2020, Potsdam, Germany, March 9–11, 2020. Zentrale Tracks, pp. 1844–1856. GITO (2020). https://doi.org/10.30844/wi_2020_r13-schindelmann
218. Schmidt, D.C.: Guest editor's introduction: model-driven engineering. Computer **39**(2), 25–31 (2006)
219. Scrocca, M., Comerio, M., Carenini, A., Celino, I.: Modelling business agreements in the multimodal transportation domain through ontological smart contracts. CoRR (2022). https://doi.org/10.48550/ARXIV.2209.05463
220. Seebacher, S., Maleshkova, M.: A model-driven approach for the description of blockchain business networks. In: Hawaii International Conference on System Sciences. HICSS (2018). https://doi.org/10.24251/HICSS.2018.442
221. Sergey, I., Kumar, A., Hobor, A.: Scilla: a smart contract intermediate-level language. CoRR (2018). https://doi.org/10.48550/ARXIV.1801.00687
222. Shi, Y., Ying, J., Shi, D., Yan, J.: Service-oriented modeling for blockchain-enabled supply chain quality information systems. Secur. Commun. Netw. **2022**, 1–16 (2022). https://doi.org/10.1155/2022/1987933
223. Silva, D., Guerreiro, S., Sousa, P.: Decentralized enforcement of business process control using blockchain. In: Advances in Enterprise Engineering XII—8th Enterprise Engineering Working Conference, EEWC 2018, Luxembourg, May 28–June 1, 2018, Proceedings. Lecture Notes in Business Information Processing, vol. 334, pp. 69–87. Springer (2018). https://doi.org/10.1007/978-3-030-06097-8_5
224. Singh, A., Parizi, R.M., Zhang, Q., Choo, K.R., Dehghantanha, A.: Blockchain smart contracts formalization: approaches and challenges to address vulnerabilities. Comput. Secur. (2020). https://doi.org/10.1016/j.cose.2019.101654
225. Six, N., Herbaut, N., Salinesi, C.: Harmonica: a framework for semi-automated design and implementation of blockchain applications. INSIGHT **24**(4), 25–27 (2021). https://doi.org/10.1002/inst.12358
226. Skotnica, M., Pergl, R.: Das contract—a visual domain specific language for modeling blockchain smart contracts. In: Advances in Enterprise Engineering XIII—9th Enterprise Engineering Working Conference, EEWC 2019, Lisbon, Portugal, May 20–24, 2019, Revised Papers. Lecture Notes in Business Information Processing, vol. 374, pp. 149–166. Springer (2019). https://doi.org/10.1007/978-3-030-37933-9_10
227. Skotnica, M., Klicpera, J., Pergl, R.: Towards model-driven smart contract systems—code generation and improving expressivity of smart contract modeling. In: Proceedings of the 20th CIAO! Doctoral Consortium, and Enterprise Engineering Working Conference Forum 2020. CEUR Workshop Proceedings, vol. 2825. CEUR (2020)
228. Skotnica, M., Aparício, M., Pergl, R., Guerreiro, S.: Process digitalization using blockchain: EU parliament elections case study. In: Proceedings of the 9th International Conference on Model-Driven Engineering and Software Development, MODELSWARD 2021, Online Streaming, February 8–10, 2021, pp. 65–75. ScitePress (2021). https://doi.org/10.5220/0010229000650075
229. Spalazzi, L., Spegni, F., Corneli, A., Naticchia, B.: Blockchain based choreographies: the construction industry case study. Concurr. Comput. Pract. Exp. (2021). https://doi.org/10.1002/cpe.6740
230. Sterling, L.S., Taveter, K.: The Art of Agent-Oriented Modeling. MIT Press, Cambridge (2009). https://doi.org/10.7551/mitpress/7682.001.0001
231. Sturm, C., Szalanczi, J., Schönig, S., Jablonski, S.: A lean architecture for blockchain based decentralized process execution. In: Business Process Management Workshops—BPM 2018 International Workshops, Sydney, NSW, Australia, September 9–14, 2018, Revised Papers. Lecture Notes in Business Information Processing, vol. 342, pp. 361–373. Springer (2018). https://doi.org/10.1007/978-3-030-11641-5_29
232. Sturm, C., Scalanczi, J., Schönig, S., Jablonski, S.: A blockchain-based and resource-aware process execution engine. Future Gener. Comput. Syst. **100**, 19–34 (2019). https://doi.org/10.1016/j.future.2019.05.006
233. Sturm, C., Szalanczi, J., Jablonski, S., Schönig, S.: Decentralized control: A novel form of interorganizational workflow interoperability. In: The Practice of Enterprise Modeling—13th IFIP Working Conference, PoEM 2020, Riga, Latvia, November 25–27, 2020, Proceedings. Lecture Notes in Business Information Processing, vol. 400, pp. 261–276. Springer (2020). https://doi.org/10.1007/978-3-030-63479-7_18
234. Suvorov, D., Ulyantsev, V.: Smart contract design meets state machine synthesis: case studies. CoRR (2019). https://doi.org/10.48550/ARXIV.1906.02906
235. Syahputra, H., Weigand, H.: The development of smart contracts for heterogeneous blockchains. In: Enterprise Interoperability VIII: Smart Services and Business Impact of Enterprise Interoperability, Proceedings of I-ESA 2018, Berlin, Germany, 2018. Proceedings of the I-ESA Conferences, vol. 9, pp. 229–238. Springer (2018). https://doi.org/10.1007/978-3-030-13693-2_19
236. Tan, S., Bhowmick, S.S., Chua, H., Xiao, X.: LATTE: Visual construction of smart contracts. In: Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, Online Conference [Portland, OR, USA], June 14-19, 2020, pp. 2713–2716. ACM (2020). https://doi.org/10.1145/3318464.3384687
237. Teruel, M.A., Trujillo, J.: Easing DApp interaction for non-blockchain users from a conceptual modelling approach. Appl. Sci. **10**(12), 4280 (2020). https://doi.org/10.3390/app10124280
238. The Object Management Group® (OMG): Business Process Model and Notation Specification (2014a). https://www.omg.org/spec/BPMN. Last access 19 Oct. 2022
239. The Object Management Group® (OMG): Object Constraint Language Specification (2014b). https://www.omg.org/spec/OCL/. Last access 19 Oct. 2022
240. The Object Management Group® (OMG): Interaction Flow Modeling Language Specification (2015a). https://www.omg.org/spec/IFML. Last access 19 Oct. 2022
241. The Object Management Group® (OMG): Unified Modeling Language (UML) (2015b). http://www.omg.org/spec/UML/. Last access 19 Oct. 2022
242. The Object Management Group® (OMG): Case Management Model and Notation Specification (2016). https://www.omg.org/spec/CMMN. Last access 19 Oct. 2022
243. The Object Management Group® (OMG): Decision Model and Notation Specification (2022). https://www.omg.org/spec/DMN. Last access 19 Oct. 2022
244. The Open Group: ArchiMate® 3.1 Specification (2022). https://pubs.opengroup.org/architecture/archimate3-doc/. Last access 19 Oct. 2022
245. Tisi, M., Mottu, J.-M., Kolovos, D., De Lara, J., Guerra, E., Di Ruscio, D., Pierantonio, A., Wimmer, M.: Lowcomote: training the next generation of experts in scalable low-code engineering platforms. In: STAF 2019. CEUR Workshop Proceedings, vol. 2405. CEUR (2019)

246. Tolmach, P., Li, Y., Lin, S., Liu, Y., Li, Z.: A survey of smart contract formal specification and verification. ACM Comput. Surv. **54**(7), 148–114838 (2022). https://doi.org/10.1145/3464421
247. Tonga Naha, R., Zhang, K.: Pupa: Smart contracts for BPMN with time-dependent events and inclusive gateways. In: Business Process Management: Blockchain, Robotic Process Automation, and Central and Eastern Europe Forum, vol. 459, pp. 21–35. Springer (2022). https://doi.org/10.1007/978-3-031-16168-1_2
248. Torres, W., van den Brand, M.G.J., Serebrenik, A.: A systematic literature review of cross-domain model consistency checking by model management tools. Softw. Syst. Model. **20**(3), 897–916 (2021). https://doi.org/10.1007/s10270-020-00834-1
249. Tran, A.B., Lu, Q., Weber, I.: Lorikeet: A model-driven engineering tool for blockchain-based business process execution and asset management. In: Proceedings of the Dissertation Award, Demonstration, and Industrial Track at BPM 2018 Co-located with 16th International Conference on Business Process Management (BPM 2018), Sydney, Australia, September 9–14, 2018. CEUR Workshop Proceedings, vol. 2196, pp. 56–60. CEUR (2018)
250. Trebbau, S., Wizenty, P., Sachweh, S.: Towards integrating blockchains with microservice architecture using model-driven engineering. In: Agile Processes in Software Engineering and Extreme Programming–Workshops—XP 2021 Workshops, Virtual Event, June 14–18, 2021, Revised Selected Papers. Lecture Notes in Business Information Processing, vol. 426, pp. 167–175. Springer (2021). https://doi.org/10.1007/978-3-030-88583-0_16
251. Tsai, W., Ge, N., Jiang, J., Feng, K., He, J.: Invited paper: Beagle: A new framework for smart contracts taking account of law. In: 13th IEEE International Conference on Service-Oriented System Engineering, SOSE 2019, San Francisco, CA, USA, April 4–9, 2019. IEEE (2019). https://doi.org/10.1109/SOSE.2019.00028
252. Tsiounis, K., Kontogiannis, K.: Goal and policy based code generation and deployment of smart contracts. In: IEEE International Conference on Software Analysis, Evolution and Reengineering, SANER 2022, Honolulu, HI, USA, March 15–18, 2022, pp. 1227–1230. IEEE (2022). https://doi.org/10.1109/SANER53432.2022.00145
253. Udokwu, C., Norta, A.: Deriving and formalizing requirements of decentralized applications for inter-organizational collaborations on blockchain. Arab. J. Sci. Eng. **46**(9), 8397–8414 (2021). https://doi.org/10.1007/s13369-020-05245-4
254. Udokwu, C., Anyanka, H., Norta, A.: Evaluation of approaches for designing and developing decentralized applications on blockchain. In: Proceedings of the 2020 4th International Conference on Algorithms, Computing and Systems, pp. 55–62. ACM (2020). https://doi.org/10.1145/3423390.3426724
255. Udokwu, C., Brandtner, P., Norta, A., Kormiltsyn, A., Matulevičius, R.: Implementation and evaluation of the DAOM framework and support tool for designing blockchain decentralized applications. Int. J. Inf. Technol. **13**(6), 2245–2263 (2021). https://doi.org/10.1007/s41870-021-00816-6
256. van den Heuvel, W., Tamburri, D.A., D'Amici, D., Izzo, F., Potten, S.: ChainOps for smart contract-based distributed applications. In: Business Modeling and Software Design—11th International Symposium, BMSD 2021, Sofia, Bulgaria, July 5–7, 2021, Proceedings. Lecture Notes in Business Information Processing, vol. 422, pp. 374–383. Springer (2021). https://doi.org/10.1007/978-3-030-79976-2_25
257. Varela-Vaca, Á.J., Quintero, A.M.R.: Smart contract languages: a multivocal mapping study. ACM Comput. Surv. **54**(1), 3–1338 (2021). https://doi.org/10.1145/3423166
258. Vingerhoets, A.S., Heng, S., Wautelet, Y.: Using i* and UML for blockchain oriented software engineering: Strengths, weaknesses, lacks and complementarity. Complex Syst. Inform. Model. Q. **26**, 26–45 (2021). https://doi.org/10.7250/csimq.2021-26.02
259. Vingerhouts, A.S., Heng, S., Wautelet, Y.: Organizational modeling for blockchain oriented software engineering with extended-i* and UML. In: Proceedings of the Workshops Co-organized with the 13th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modelling (PoEM 2020), On-line (originally Located in Riga, Latvia), November 26, 2020. CEUR Workshop Proceedings, vol. 2749, pp. 23–34. CEUR (2020)
260. vom Brocke, J., Simons, A., Riemer, K., Niehaves, B., Plattfaut, R., Cleven, A.: Standing on the shoulders of giants: Challenges and recommendations of literature search in information systems research. Commun. Assoc. Inf. Syst. **37**, 9 (2015)
261. W3C OWL Working Group: OWL 2 Web Ontology Language Document Overview (Second Edition) (2012). https://www.w3.org/TR/owl2-overview/. Last access 19 Oct. 2022
262. Watson, R.T., Webster, J.: Analysing the past to prepare for the future: Writing a literature review, a roadmap for release 2.0. J. Decis. Syst. **29**(3), 129–147 (2020)
263. Weber, I., Xu, X., Riveret, R., Governatori, G., Ponomarev, A., Mendling, J.: Untrusted business process monitoring and execution using blockchain. In: Business Process Management—14th International Conference, BPM 2016, Rio de Janeiro, Brazil, September 18–22, 2016. Proceedings. Lecture Notes in Computer Science, vol. 9850, pp. 329–347. Springer (2016). https://doi.org/10.1007/978-3-319-45348-4_19
264. Webster, J., Watson, R.T.: Analyzing the past to prepare for the future: writing a literature review. MIS Q. **26**(2), 11 (2002)
265. Weingärtner, T., Rao, R., Ettlin, J., Suter, P., Dublanc, P.: Smart contracts using Blockly: Representing a purchase agreement using a graphical programming language. In: Crypto Valley Conference on Blockchain Technology, CVCBT 2018, Zug, Switzerland, June 20–22, 2018, pp. 55–64. IEEE (2018). https://doi.org/10.1109/CVCBT.2018.00012
266. Whittle, J., Hutchinson, J., Rouncefield, M.: The state of practice in model-driven engineering. IEEE Softw. **31**(3), 79–85 (2013)
267. Wickramarachchi, V.U., Keppitiyagama, C.I., Gunawardana, K.G.: Efficiently transform contracts written in Peyton Jones contract descriptive language to Solidity. In: 2019 19th International Conference on Advances in ICT for Emerging Regions (ICTer), pp. 1–8. IEEE (2019). https://doi.org/10.1109/ICTer48817.2019.9023652
268. Wieland, M., Fill, H.-G.: A domain-specific modeling method for supporting the generation of business plans. In: Bork, D., Karagiannis, D., Mayr, H.C. (eds.) Modellierung 2020, 19–21. Februar 2020, Wien, Österreich. LNI, vol. P-302, pp. 45–60. Gesellschaft für Informatik e.V (2020)
269. Winter, R., Schelp, J.: Reference modeling and method construction: a design science perspective. In: Haddad, H. (ed.) Proceedings of the 2006 ACM Symposium on Applied Computing (SAC), Dijon, France, April 23–27, 2006, pp. 1561–1562. ACM (2006). https://doi.org/10.1145/1141277.1141638
270. Wöhrer, M., Zdun, U.: Domain specific language for smart contract development. In: IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2020, Toronto, ON, Canada, May 2–6, 2020, pp. 1–9. IEEE (2020a). https://doi.org/10.1109/ICBC48266.2020.9169399
271. Wöhrer, M., Zdun, U.: From domain-specific language to code: smart contracts and the application of design patterns. IEEE Softw. **37**(4), 37–42 (2020). https://doi.org/10.1109/MS.2020.2993470
272. Wood, D.G.: Ethereum: A secure decentralized generalized transaction ledger. Yellowpaper (2022). https://ethereum.github.io/yellowpaper/paper.pdf
273. Yao, L., Mimno, D., McCallum, A.: Efficient methods for topic model inference on streaming document collections. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '09, pp. 937–946. ACM (2009). https://doi.org/10.1145/1557019.1557121

274. Yu, E.S.K.: Modeling strategic relationships for process reengineering. PhD thesis, University of Toronto, Dept. of Computer Science, Toronto (1995)
275. Yu, E.S.K., Giorgini, P., Maiden, N., Mylopoulos, J.: Social Modeling for Requirements Engineering. MIT Press, Cambridge (2010). https://doi.org/10.7551/mitpress/7549.001.0001
276. Yu, E.S.K., Mylopoulos, J.: Understanding "why" in software process modelling, analysis, and design. In: Proceedings of 16th International Conference on Software Engineering, pp. 159–168. IEEE (1994). https://doi.org/10.1109/ICSE.1994.296775
277. Zupan, N., Kasinathan, P., Cuellar, J., Sauer, M.: Secure smart contract generation based on Petri Nets. In: Blockchain Technology for Industry 4.0, pp. 73–98. Springer (2020). https://doi.org/10.1007/978-981-15-1137-0_4

Simon Curty is a PhD student and assistant in the Digitalization and Information Systems Group at the University of Fribourg, Switzerland. His main areas of research include distributed ledger technologies, enterprise architecture, model-driven engineering, and conceptual modeling. Of specific interest are domain-specific languages and conceptual modeling methods for the development of blockchain-based applications. Before starting his doctoral research, he obtained an M.Sc. in computer science from the University of Bern and worked in industry as a software developer and engineer.

Felix Härer is a senior researcher and lecturer in the Digitalization and Information Systems Group at the University of Fribourg, Switzerland. Following his PhD on blockchains and decentralized organizations, received from the University of Bamberg, Germany (2019), his main areas of research include blockchains, cloud computing, conceptual modeling, software and requirements engineering, data management, and data analytics. Before, he was an assistant at the University of Fribourg (2018–2020) and the University of Bamberg (2014–2018), where he worked in the System Development and Database Application (SEDA) Group, was working in industry, e.g., for Siemens Healthcare, and in various research projects.

Hans-Georg Fill is a full professor for business informatics and head of the research group Digitalization and Information Systems at the University of Fribourg, Switzerland. He holds a Ph.D. and a habilitation from the University of Vienna, Austria. His research areas include metamodeling, digitalization, blockchains, and augmented and virtual reality. In 2022 he was elected as speaker of the Special Interest Group on Modeling Enterprise Information Systems (GI-MobIS) in the German Informatics Society.