APPENDICES

Appendix A: Details on the Research Method to Identify and Synthesize Smart Contract Development Challenges and Corresponding Solutions

We performed four iterations of data gathering accompanied by iterative data analyses to achieve theoretical saturation. In the following, each iteration will be described in more detail to enhance method transparency and reproducibility as well as show method rigor.

A.1 First Iteration: Snowballing Literature Review

To generate an initial set of smart contract challenges and solutions as a foundation for our study, we conducted a literature search following the snowballing approach proposed by Wohlin [1]. Snowballing starts from a core set of relevant documents and then uses these to identify further relevant documents in a specific domain through multiple iterations of forward- and backward-snowballing.

Snowballing Literature Search [1]. We first defined the search string *"Smart Contract*" AND ("Challeng*" OR "Vulnerabilit*")* and applied it to Google Scholar as suggested by Wohlin [1]. The search string yielded 879 documents as of 4th July 2019. We exported each document's meta-information (e.g., title, authors, year, and abstract) into a spreadsheet to perform a relevancy check. We analyzed this meta-information to identify the documents relevant to answer our research question by applying inclusion and exclusion criteria (see Table A1). In particular, we included documents that name, propose or discuss smart contract challenges or solutions to overcome challenges, including SDPs. We excluded all documents that were not in English, not related to our topic (e.g., dealing with the use of smart contracts but not describing their implementation), books, or duplicates. *Please Note:* If we identified a document published in multiple occurrences (e.g., at conferences and in journals), we carefully compared its content, looked for improvements or corrections in later versions, and synthesized the findings [2].

While conducting the snowballing literature search, we also identified grey literature, such as blog entries, DLT- and smart-contract-related documentation, company whitepapers, and preprints (e.g., from *arxiv.org*), that is not formally published in scientific sources (e.g., conference proceedings or journal articles). We included this grey literature due to three reasons. First, potential and novel smart contract challenges may be revealed and discussed in the non-scientific DLT communities. Second, we aimed to develop a more sophisticated understanding of, and appreciation for, practitioners' defeasible evidence, inference, and belief [3]. Third, we strived to benefit from synthesizing both research and industry knowledge by revealing consensus or, more importantly, contradictions between them. We followed extant guidelines for including grey literature in literature reviews (e.g., applying the criteria "authority of the producer, methodology, objectivity, date, novelty, impact, as well as outlet control" to assess the quality of the grey literature [4, p. 12]).

TABLE A1
INCLUSION AND EXCLUSION CRITERIA FOR LITERATURE SEARCHES IN ITERATION 1 AND ITERATION 3

Type	Name	Description
Inclusion Criteria	Challenge	Naming, proposing, discussing, or revealing smart contract challenges, vulnerabilities, or related issues
	Solution	Naming, proposing, discussing, or revealing solutions addressing smart contract challenges, vulnerabilities, or related issues
	Concrete SDP	Naming, proposing, discussing, or revealing SDPs relating to smart contracts
	Transferrable SDP	Naming, proposing, discussing, or revealing SDPs that are applicable or transferable to smart contracts
Exclusion	Off-topic	Not dealing with DLT and smart contracts
	Books	Books on smart contracts
	Not in English	Publications in a non-English language
	Duplicates	Multiple identical occurrences of a document

After applying the inclusion and exclusion criteria, we started with a preliminary set of 21 potentially relevant candidate documents. Next, we carefully read the candidate documents' full texts to ensure that we include only those documents in our starting set that are particularly relevant to answering the research question. We concluded with a set of ten relevant documents in the starting set on which we conducted the initial iteration of backward and forward search [1]. For the backward search, we studied the references of the starting set to identify more relevant documents for our study. For the forward search, we searched for documents that cite those of our starting set using the citation sources on Google Scholar. To assess the relevance of the 778 candidate documents found in the backward (119) and forward (659) search, we again exported the meta-information for each document into a spreadsheet and applied the same inclusion and exclusion criteria as before, and finally added ten documents to the start set.

We similarly conducted the second iteration of the backward and forward search, considering the ten new documents from the first iteration. We identified 58 new candidate documents, where we included one document after applying the exclusion criteria. Afterward, we conducted the third iteration of the backward and forward search, revealing 25 candidate documents that we deemed irrelevant to our study. Since we could not identify additional documents relevant to

our topic, we deem the snowballing literature search finished [1]. Through our first iteration, we thus identified 21 relevant documents in total to be analyzed.

Literature Analysis. We applied thematic analysis Feld [5] for the literature analysis to identify themes for challenges and related solutions apparent in smart contract development. The thematic analysis includes six phases [5]: familiarize yourself with the data, generate initial codes, search for themes, review themes, define and name themes, and produce the report.

In the first phase (*familiarize yourself with the data*), we listed abstracts, keywords, publication years, examined ledgers, and programming languages, among others, of each study to gain a general impression of the relevant documents. For example, the documents predominantly focus on Ethereum smart contracts developed in the programming language Solidity (e.g., [6], [7]).

In the second phase (*generate initial codes*), we coded the relevant documents to extract preliminary challenges and solutions in smart contract development. We performed data-driven coding in two rounds of refinement. In the first coding round, we read the documents in detail, extracted potential text segments about challenges and solutions, and then assigned them either a challenge or solution label as a sub-theme. We coded 146 text segments relating to 75 challenges and 71 solutions. For example, we identified the vulnerability *Call to the unknown* [8] and mapped it to challenge, and assigned the proposed *Contract Observer Pattern* [9] to a solution. In the second coding round, we analyzed the text segments in detail to identify the specifics of each challenge or solution and assign them an appropriate label. For example, for the text segment "In classical re-entrance attacks, the same function of the contract is re-entered again. […]" [10, p. 5], we determined the label *reentrancy* as one development challenge.

In the third phase (*search for themes*), we again analyzed our codes and related text segments in detail and considered how different codes may combine to form an overarching theme [5]. Since different people often put the same labels on different things and vice versa, it is crucial for the validity of qualitative analysis to be aware of potential semantic ambiguities (e.g., same terminology for different concepts) [11]. For example, we merged the labels *access control* and *permissioning* to the sub-theme *access control*. By constantly comparing our initial codes and the respective text segments, we were able to identify 18 candidate themes, such as *exception handling* and *event order*.

We then moved on to the fourth phase (*review themes*) to further refined the candidate themes. We reviewed our candidate themes and respective text segments while applying Patton's [12] dual criteria of internal homogeneity (i.e., data within themes should cohere together meaningfully) and external heterogeneity (i.e., there should be clear and identifiable distinctions between themes). We first merged challenge and solution sub-themes since identified solutions relate to specific challenges and can therefore be assigned to the same sub-theme. We then collated the 18 candidate themes into seven themes (e.g., *authorization, data type,* and *maintainability*). For example, we assigned *event order* to the theme *program flow*.

In the fifth phase (*define and name themes*), we captured the essence of each theme. Based on these findings, we developed a definition of each theme and sub-theme. In the sixth phase (*produce the report*), we wrote detailed descriptions of each of the seven themes as a foundation for the next data gathering and analysis phases.

A.2 Second Iteration: Focus Group Interview

After completing the snowballing search and thematic analyses, we conducted a second iteration of data gathering and analyses for two reasons. First, we wanted to validate our literature findings because qualitative coding techniques bear the risk of interpretation biases. Second, we strived to incorporate knowledge and experiences from experts in the field to extend and enrich our themes. We, therefore, decided to conduct a focus group interview with DLT experts.

Focus Group Interview [13]. By conducting focus group interviews, one can collect viewpoints about a certain defined topic of interest from a group of people who have certain experiences [13]. Focus groups also enable participants to engage in thoughtful discussions and generate valuable and extensive data. We conducted the focus group interview in July 2019 using a convenience sample of five DLT experts (see Table A2). Three researchers participated in and moderated the workshop.

We conducted the focus group interview based on an interview guide [14]. The interview guide kept the interactions focused while allowing individual experiences to emerge during the limited interview period [15]. The interview guide served as a reminder of the information that we needed to collect [14]. Specifically, we began the interview with an introduction, a short explanation of our research, and some demographic questions about the interviewees. Next, we started a brainstorming phase about potential challenges in smart contract development. We asked each participant to write down at least three challenges to be presented and discussed in the group. We used a flip chart to keep and organize the intermediate results. Afterward, we discussed whether and how the challenges on the flip chart can be further grouped based on similarities. We also asked participants about potential solutions for identified challenges and noted them down. After this open discussion phase, we asked questions that were based on our seven themes identified from the first iteration to validate them and gather additional data based on the participants' opinions. The interview lasted six hours and was recorded and then transcribed.

During the focus group interview, we applied a nonjudgmental form of listening [16], maintained distance [12], and strived to maintain an open and non-directive style of conversation to ensure impartiality [17]. We applied projective techniques to uncover participants' innermost thoughts and feelings, particularly by relating our questions to previous

DLT projects they were involved in [18]. We have also been sensitive to ethical concerns by (1) balancing anonymity and disclosure [19]; (2) ensuring that transcripts and other data were kept secure [20]; (3) respecting participants' knowledge and time (i.e., we scheduled meetings to fit the interviewees' schedules and frequently acknowledged their efforts) [20].

TABLE A2
OVERVIEW OF INTERVIEWEES' DEMOGRAPHICS

	Iteration 2	Iteration 4
Number of Experts	5	9
DLT Expertise	Ethereum	Ethereum, EOSIO, Hyperledger Fabric
Average Software Engineering Experience, in years	3.9	9.1
Average Smart Contract Development Experience, in years	2.9	3.2
Industry	Energy, IT	Automotive, Energy, IT

Interview Data Analysis [21]. We applied scientific coding techniques to analyze the interview data, especially selective, open, and axial coding [21]. We started with selective coding by assigning themes identified in the literature review to the interview findings to validate findings from prior research and gather additional information. We thus inserted the themes into our codebook and selectively assigned these codes to textual segments of the interview transcripts that relate to the specific theme. Afterward, we performed open coding to identify new challenges and solutions that have been neglected in prior research so far. Open coding entails fracturing the data by describing concepts that may define a significant occurrence or incident about a phenomenon [21]. During open coding, all available data were labeled for direct visibility of the structure and information of the interview. Finally, we used axial coding to identify the causes and consequences of each challenge [21], [22]. Causes refer to the perceived reasons that persons give for why things happen. Consequences refer to the outcomes of actions. With axial coding, we could understand why challenges appear and what the potential consequences of these challenges are.

Through these coding steps, we identified one new challenge theme and a corresponding solution (i.e., code discoverability), refined existing themes (e.g., enriching their descriptions about causes and consequences), identified minor inconsistencies in the descriptions of the challenges and solutions in smart contract development and resolved these inconsistencies accordingly (e.g., we unified the levels of abstraction of the solutions). Finally, we updated our report on challenges and solutions (in line with the sixth phase of thematic analyses) as a foundation for the next data-gathering and analysis phases.

A.3 Third Iteration: Database Literature Review

We decided to turn back to scientific literature in the next data-gathering and analysis iteration for two reasons. First, new documents may have been published since our first literature search. Second, we might have overlooked relevant documents given our snowballing research approach. We, therefore, performed a systematic literature search in scientific databases, ultimately trying to achieve theoretical saturation.

Systematic Literature Search in Scientific Databases [23]. Our descriptive literature review [24] was guided by recommendations for reviews in the software engineering domain [23]. For the identification of documents addressing smart contract challenges and solutions, we searched scientific databases that we deemed representative as they cover a wide range of journal articles as well as conference papers from computer science, information systems, and related disciplines: ACM Digital Library, Association for Information Systems Electronic Library (AISel), EBSCOHost, IEEEXplore, Proquest (ABI/INFORM), ScienceDirect, and Web of Science. To cover a broad set of documents, potentially relevant papers needed to meet the *("smart contract*" OR "chaincode") AND ("challenge*" OR "pattern*" OR "issue*" OR "develop*" OR "programming")* search string in the title, abstract, or keywords (TAK). We limited our search to peer-reviewed documents (when possible) to ensure the high quality of documents. Our search yielded 1774 potentially relevant documents as of 6th November 2020.

To filter documents, we first checked the relevancy of each document by analyzing TAK. If any indication of relevancy appeared, the document was marked for further processing. We applied our first iteration's inclusion and exclusion criteria (see Table A1). This first TAK relevancy assessment resulted in a sample of 263 documents deemed to be potentially relevant. Afterward, we performed a fine-grained relevance validation by reading the documents in detail, resulting in a sample of 96 relevant documents. In particular, we excluded documents that were duplicates (248), not a research paper (e.g., editorials and dissertations) and books (28), not in English (1), or off-topic (e.g., dealing with smart contracts but not with challenges or solutions; 1401). Comparing relevant documents with the set of literature from the first iteration revealed that ten of the 21 relevant papers from the first iteration were also included in the set of the database literature review, leading to a final set of 86 novel documents that we then analyzed.

Literature Analysis [5]. We again applied thematic analysis [5] to refine our existing and identify novel themes for challenges and solutions in the relevant documents. We first familiarized ourselves with the new literature, for example, by noting the analyzed DLT protocol and topic of the documents.

Second, we performed data-driven coding to extract challenges and solutions in two rounds of refinement. In the first coding round, we read the documents in detail, extracted potential text segments related to challenges and solutions, and then assigned them either a *challenge* or *solution* label as a sub-theme. In total, we coded 1,018 text segments, of which

we classified 823 as related to challenges and 195 as about solutions. In the second coding round, we analyzed the text segments in detail to identify the specifics of each challenge or solution and assign them an appropriate label. For example, for the text segment "How to verify the logic of contract is correct and how to eliminate the loopholes in the contract?" [25, p. 323], we determined the label *correctness* as one challenge. By grouping similar labels, we identified 268 challenges and 89 solutions in total.

In the third phase, we reviewed the text segments and aggregated individual challenges and solutions to identify initial sub-themes. We first assigned our existing themes to the text segments, if possible. Otherwise, we created a new sub-theme. By comparing our initial sub-themes and the respective text segments, we aggregated our labels to 83 challenge and 39 solution sub-themes, such as *redundant instruction*, *gas consumption*, and *dead code*. Since we still faced many sub-themes, we conducted another iteration of coding in which we further grouped the sub-themes according to their similarities and distinctions with other sub-themes. This fourth round of coding resulted in 53 candidate themes.
We then moved on to the review themes phase to refine these candidate themes further. We reviewed our candidate themes and respective text segments. We collated the 53 candidate themes into 13 themes (e.g., *code efficiency, confidentiality*, and *determinism*). For example, we assigned *required interactions* to the theme *code efficiency* while we assigned *code visibility* to the theme *confidentiality*. With this final coding step, we achieved higher theoretical abstraction. To further group these themes, we compared the different origins for each challenge theme and strived to identify a core set of common origins. Our comparative analysis revealed three principal challenge origins that can cause flaws in smart contract code (see Table A3): *platform*, *programming language & execution environment*, and *coding practice.*

TABLE A3
PRINCIPAL CHALLENGE ORIGINS THAT CAN CAUSE CHALLENGES IN SMART CONTRACT DEVELOPMENT

Component	Description
Platform	The protocol put in place to manage the interactions between nodes and define the procedures for the issuance, verification, and storage of transactions
Programming Language & Execution Environment	The capabilities offered to develop smart contracts and execute them via the distributed ledger
Coding Practice	The development activities to achieve a specified outcome in the form of a smart contract

Finally, we captured the essence of what each theme is about and defined each theme and sub-theme. Compared to our first snowballing literature research, we identified six novel high-level themes and refined existing themes. We created a hierarchy of themes ranging from text segments, initial challenge and solution labels up to aggregated challenges and solutions assigned to sub-themes that we grouped into three principal challenge origins.

A.4 Fourth Iteration: Expert Interviews

While we had already gathered rich information on various challenges and were able to identify solutions for many of them, we decided to conduct a fourth iteration due to two reasons. First, we strived to validate our literature review findings with further knowledge and experience of DLT experts. Second, most of the literature focuses on Ethereum, and the EVM and Solidity, respectively. We were eager to reflect critically on the applicability of our findings to other DLT protocols. We focused on EOSIO and Hyperledger Fabric besides Ethereum because these DLT protocols are frequently used in organizational contexts and allow for insights into three strongly differing smart contract integration concepts. Therefore, in the fourth iteration, we conducted semi-structured interviews with DLT experts.

Expert Interviews [13]. To acquire interviewees, we contacted potential interviewees in multiple ways. First, we relied on personal contacts from previous and ongoing DLT research projects (i.e., a convenience sample). We acquired three DLT researchers and one practitioner having knowledge about and experience with Ethereum smart contracts. Second, we emailed smart contract developers on GitHub who worked with Hyperledger Fabric or EOSIO and fulfilled certain quality criteria (e.g., we assessed the number and frequency of commits). We also emailed the authors of research documents related to our study. Besides, we contacted developers in well-known companies working on smart contract development. In total, we sent about 50 invitations to participate in our study, from which five developers accepted our invitation. In total, we were thus able to conduct nine interviews. The interviewees had an average experience of 9.1 years in general software engineering and 3.2 years in smart contract development by the time of interviews (see Table A2). Besides, they held various roles, including developers, testers, project managers, architects, designers, CEO/CTO, research assistants, and smart contract trainers. This diverse group of practitioners helped us to reflect critically on our challenges and solutions and understand the specifics of DLT protocols, programming languages, and execution environments.

Again, we prepared an interview guide Feld [14] and a slide deck to structure the interview and ask questions based on identified challenges and solutions. We began each interview with an introduction, a short explanation of our research, and demographic questions about the interviewees. We then explained each challenge and the potential solution in detail and asked participants to comment and critically discuss our intermediate findings freely. In particular, we

asked whether these challenges and solutions are also relevant and applicable to other ledgers. We also asked open questions about whether the interviewees perceive further challenges or have novel solutions for our challenges that we have missed or may be specific to DLT protocols. Since the interviews were semi-structured, we attentively asked follow-up questions to dig deeper into our interviewees' viewpoints when appropriate [26]. At the end of the interview, we asked the interviewee to provide any other important information we may have missed. After each interview, we adapted the interview guide if new challenges or solutions emerged and to validate prior findings.

All interviews were performed remotely via video conference tools and recorded with the participants' permission. Interviews were scheduled between December 2020 and February 2021. The average interview time was 63 minutes. The researcher who carried out the interview took notes on each challenge and solution discussed with the interview partner during the interview.

Interview Data Analysis. Taking notes enabled us to reduce interpretation bias and directly resolve open questions. After each interview, we analyzed the field notes and compared them to our intermediate findings from the third iteration. We enriched our theme descriptions, renamed themes and challenges suggested by the interviewees, and added novel challenges and solutions and ledger-specifics to each challenge and solution description. If any ambiguity or unclarity appeared, the authors first turned to the audio interview recordings or contacted the interview partner for clarification.

Interviewees also recommended specific documents we may incorporate into our analyses (e.g., grey literature, research articles, ledger documentation), which we considered after the interviews. Notably, several interviewees offered us the opportunity to review this work, leading to minor improvements and corrections.

Analysis results validated the three principal challenge origins and led to refinements of sub-theme challenges. More importantly, we were able to identify novel challenges and solutions by comparing challenges and solutions across ledgers and considering ledger-specifics. For example, we identified the challenge of *Non-deterministic Behavior* that only applies to smart contracts developed in Go. By finishing the fourth iteration, we identified 29 challenges and 60 solutions.

To ensure that we identified a reliable set of challenges and solutions, we followed researchers stressing that an important goal is to reach theoretical saturation in qualitative research [27]–[29]. Theoretical saturation is often taken to indicate that further data collection or analysis is unnecessary based on the data analyzed hitherto because it is unlikely that additional data collection will generate new findings [30], [31]. Lincoln and Guba [32] speak of the term *point of redundancy* in this context.

We first looked at our literature review protocols, revealing that our literature analysis did not reveal new challenges or solutions since the last 12 analyzed documents. Similarly, we asked our interviewees during our fourth iteration if they knew of any further challenges or solutions that have yet to be discussed during the interviews. Besides, we sent our manuscript to the interviewees after finishing the writing to ask them once more if they had any challenges or solutions to add. In both cases, interviewees agreed that they are not aware of further challenges and solutions to the best of their knowledge. Consequently, we are confident to have reached (at least) a sufficient degree of theoretical saturation after completing our fourth iteration and thus refrained from performing additional iterations of data gathering and analyses. We acknowledge that smart contract development is a fast-evolving field; therefore, further challenges and solutions may appear in the (near) future.

Appendix B: Evaluation Criteria for Software Design Patterns

Group	Criterion	Definition	Applied	Exemplary Reference
Flex.	Adaptability	The degree to which the SDP can be combined with other SDPs to solve a problem that could not be solved by the original SDP alone should be high.	X	[33]
	Flexible	The degree to which the SDP can be used in contexts other than originally intended should be high.		[34]
Out.	Correctness	The solution proposed in the SDP should work flawlessly.	X	[35]
	Reliability	Where applicable, the SDP should solve problems with the same accuracy when applied repeatedly.		[35]
Pattern Design	Completeness	The level of completeness of the description of an SDP (e.g., context and applicability) should be such that the SDP can be used unambiguously.	X	[36]
	Consistency	The used language and terminology in the SDP description should be uniform and free from contradictions.	X	[35]
	End-User Oriented	The degree to which the SDP is tailored to the needs and capabilities of the target entity that will primarily use the SDP should be high.		[34]
	Positive Framing	The SDP should explain how to solve a problem instead of describing how to cause a problem.	X	[34]
	Structure	The actual content in an SDP compared to the required pattern content (i.e., name, problem, context, forces, example, solution, resulting context, relationships to other patterns, and known uses) should be coherent.	X	[37]
Perception	Adoptability	The likeliness to which the SDP is used by developers aiming to solve a problem should be high.		[36]
	Popular Acceptance	The perceived number of usages of the SDP should increase over time.		[38]
	Relevance	The perceived importance of the SDP under consideration of, for example, the criticality and the number of occurrences of the problem to be solved by the pattern should be high.	X	[36]
	Understandability	The degree to which pattern users can comprehend the SDP without explicit prior knowledge should be high.	X	[39]
	Usefulness	The perceived helpfulness of the SDP should be high.		[40]
Utilization	Comprehensiveness	The degree of sufficient detail the SDP provides to explain to the reader when, why, and how the pattern can be applied to solve which problem, to what extent and at what cost of other characteristics should be high.	X	[35]
	Concreteness	The degree to which the SDP description does not allow for misinterpretation should be high.	X	[41]
	Debatability	The extent to which an SDP is clear enough to be criticized should be high.	X	[34]
	Pattern Use Determination	The degree to which it can be determined if the SDP has been applied to certain code or not should be high.		[34]
	Domain Independence	The degree to which the SDP applies to different application areas should be high.		[37]
	Effectiveness	The degree to which the SDP helps to improve smart contract code should be high.		[35]
	Integrability	The degree to which the SDP can be integrated into the individual software engineering processes should be high.		[37]
	Level of Abstraction	The level to which the SDP is abstracting from a precise solution for a particular occurrence of the problem should be appropriate.	X	[36]
	Usability	The ease with which developers understand and apply the SDP should be high.		[35]

X: Applied in focus group to evaluate the SDPs Flex.: Flexibility Out.: Outcome

Appendix C: Software Design Patterns

Please check our GitHub repository for the latest version of the SDP:
https://github.com/KITcii/smart-contract-dev-support

REFERENCES

[1] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *18th International Conference on Evaluation and Assessment in Software Engineering*, London, England, United Kingdom, 2014, pp. 1–10. doi: 10.1145/2601248.2601268.

[2] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," *Keele University, UK*, vol. 9, 2007.

[3] A. Rainer, "Using argumentation theory to analyse software practitioners' defeasible evidence, inference and belief," *Information and Software Technology*, vol. 87, pp. 62–80, Jul. 2017, doi: 10.1016/j.infsof.2017.01.011.

[4] V. Garousi, M. Felderer, and M. V. Mäntylä, "Guidelines for including grey literature and conducting multivocal literature reviews in software engineering," *Information and Software Technology*, vol. 106, pp. 101–121, Feb. 2019, doi: 10.1016/j.infsof.2018.09.006.

[5] V. Braun and V. Clarke, "Using thematic analysis in psychology," *Qualitative Research in Psychology*, vol. 3, no. 2, pp. 77–101, Jan. 2006, doi: 10.1191/1478088706qp063oa.

[6] M. Wöhrer and U. Zdun, "Smart contracts: security patterns in the ethereum ecosystem and solidity," in *2018 International Workshop on Blockchain Oriented Software Engineering*, Mar. 2018, pp. 2–8.

[7] M. Wöhrer and U. Zdun, "Design Patterns for Smart Contracts in the Ethereum Ecosystem," in *2018 IEEE International Conference on Internet of Things and IEEE Green Computing and Communications and IEEE Cyber, Physical and Social Computing and IEEE Smart Data*, Halifax, NS, Canada, Jul. 2018, pp. 1513–1520. doi: 10.1109/Cybermatics_2018.2018.00255.

[8] N. Atzei, M. Bartoletti, and T. Cimoli, "A Survey of Attacks on Ethereum Smart Contracts (SoK)," in *Principles of Security and Trust*, M. Maffei and M. Ryan, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2017, pp. 164–186. doi: 10.1007/978-3-662-54455-6_8.

[9] Y. Liu, Q. Lu, X. Xu, L. Zhu, and H. Yao, "Applying Design Patterns in Smart Contracts," in *Blockchain – ICBC 2018*, S. Chen, H. Wang, and L.-J. Zhang, Eds., in Lecture Notes in Computer Science, vol. 10974. Cham: Springer International Publishing, 2018, pp. 92–106. doi: 10.1007/978-3-319-94478-4_7.

[10] M. Rodler, W. Li, G. O. Karame, and L. Davi, "Sereum: Protecting Existing Smart Contracts Against Re-Entrancy Attacks," *arXiv:1812.05934 [cs]*, Dec. 2018, Accessed: Aug. 24, 2019. [Online]. Available: http://arxiv.org/abs/1812.05934

[11] M. L. G. Shaw and B. R. Gaines, "Comparing Conceptual Structures: Consensus, Conflict, Correspondence and Contrast," *Knowledge Acquisition*, vol. 1, no. 4, pp. 341–363, Dec. 1989, Accessed: Jun. 10, 2018. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S104281438980010X

[12] M. Q. Patton, *Qualitative research & evaluation methods: integrating theory and practice*, Fourth edition. Thousand Oaks, CA, USA, 2015.

[13] M. D. Myers, *Qualitative research in business & management*, 2nd ed. SAGE Publications Ltd, 2013.

[14] R. K. Yin, *Case study research: design and methods*, 4th ed. in Applied social research methods, no. 5. Los Angeles, CA, USA, 2009.

[15] R. L. Gorden, *Interviewing: strategy, techniques, and tactics*, Rev. ed. in The Dorsey series in sociology. Homewood, Ill: Dorsey Press, 1975.

[16] G. Walsham, "Interpretive case studies in IS research: nature and method," *European Journal of Information Systems*, vol. 4, no. 2, pp. 74–81, May 1995, doi: 10.1057/ejis.1995.9.

[17] H. Heath, "Exploring the influences and use of the literature during a grounded theory study," *Journal of Research in Nursing*, vol. 11, no. 6, pp. 519–528, Nov. 2006, doi: 10.1177/1744987106069338.

[18] S. Donoghue, "Projective techniques in consumer research," *JFECS*, vol. 28, no. 1, Mar. 2010, doi: 10.4314/jfecs.v28i1.52784.

[19] U. Flick, *An Introduction to Qualitative Research*, 4th ed. Los Angeles, CA, USA: SAGE, 2011.

[20] M. D. Myers and M. Newman, "The Qualitative Interview in IS Research: Examining the Craft," *Information and Organization*, vol. 17, no. 1, pp. 2–26, Jan. 2007, doi: 10.1016/j.infoandorg.2006.11.001.

[21] J. M. Corbin and A. L. Strauss, *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, 4th ed. Los Angeles, CA, USA, 2015.

[22] Strong and Volkoff, "Understanding Organization—Enterprise System Fit: A Path to Theorizing the Information Technology Artifact," *MIS Quarterly*, vol. 34, no. 4, p. 731, 2010, doi: 10.2307/25750703.

[23] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic Literature Reviews in Software Engineering," *Information and Software Technology*, vol. 51, no. 1, pp. 7–15, 2009.

[24] G. Paré, M. C. Trudel, M. Jaana, and S. Kitsiou, "Synthesizing information systems knowledge: A typology of literature reviews," *Information and Management*, vol. 52, no. 2, pp. 183–199, 2015.

[25] X. Bai, Z. Cheng, Z. Duan, and K. Hu, "Formal Modeling and Verification of Smart Contracts," in *Proceedings of the 2018 7th International Conference on Software and Computer Applications*, Kuantan Malaysia: ACM, Feb. 2018, pp. 322–326. doi: 10.1145/3185089.3185138.

[26] W. Zou *et al.*, "Smart Contract Development: Challenges and Opportunities," *IEEE Trans. Software Eng.*, pp. 1–20, 2019, doi: 10.1109/TSE.2019.2942301.

[27] G. Guest, A. Bunce, and L. Johnson, "How Many Interviews Are Enough?: An Experiment with Data Saturation and Variability," *Field Methods*, vol. 18, no. 1, pp. 59–82, Feb. 2006, doi: 10.1177/1525822X05279903.

[28] J. M. Morse, "'Data Were Saturated . . . ,'" *Qual Health Res*, vol. 25, no. 5, pp. 587–588, May 2015, doi: 10.1177/1049732315576699.

[29] P. Fusch and L. Ness, "Are We There Yet? Data Saturation in Qualitative Research," *Qualitative Report*, vol. 20, pp. 1408–1416, 2015.

[30] A. L. Strauss and J. M. Corbin, Eds., *Grounded theory in practice*. Thousand Oaks: Sage Publications, 1997.

[31] B. Saunders *et al.*, "Saturation in qualitative research: exploring its conceptualization and operationalization," *Qual Quant*, vol. 52, no. 4, pp. 1893–1907, Jul. 2018, doi: 10.1007/s11135-017-0574-8.

[32] Y. S. Lincoln and E. G. Guba, *Naturalistic inquiry*. Beverly Hills, Calif: Sage Publications, 1985.

[33] A. Dearden, J. Finlay, L. Allgar, and B. Mcmanus, "Evaluating pattern languages in participatory design," 2002, p. 664. doi: 10.1145/506486.506535.

[34] K. McGee, "Patterns and Computer Game Design Innovation," in *Proceedings of the 4th Australasian Conference on Interactive Entertainment*, in IE '07. Melbourne, AUS: RMIT University, 2007.

[35] D. Khazanchi, J. D. Murphy, and S. C. Petter, "Guidelines for Evaluating Patterns in the IS Domain," in *2nd Midwest United States Association for Information Systems Conference*, 2008. [Online]. Available: https://digitalcommons.unomaha.edu/isqafacproc/7

[36] C. Rolland, J. Stirna, N. Prekas, P. Loucopoulos, A. Persson, and G. Grosz, "Evaluating a Pattern Approach as an Aid for the Development of Organisational Knowledge: An Empirical Study," in *Active Flow and Combustion Control 2018*, R. King, Ed., in Notes on Numerical Fluid Mechanics and Multidisciplinary Design, vol. 141. Cham, 2000, pp. 176–191. doi: 10.1007/3-540-45140-4_13.

[37] J. Borchers and F. Buschmann, *A Pattern Approach to Interaction Design*. USA: John Wiley & Sons, Inc., 2001.

[38] W. Brown, R. Malveau, H. McCormick, and T. Mowbray, "The Software Patterns Criteria - Proposed Definitions for Evaluating Software Pattern Quality." Jun. 02, 1998. [Online]. Available: http://antipatterns.com/whatisapattern/

[39] D. Lea, "Christopher Alexander: an introduction for object-oriented designers," *SIGSOFT Softw. Eng. Notes*, vol. 19, no. 1, pp. 39–46, Jan. 1994, doi: 10.1145/181610.181617.

[40] P. Singpant and N. Prompoon, "Constructing patterns verification criteria based on quality attributes: Web security context patterns case study," in *2016 IEEE/ACIS 15th International Conference on Computer and Information Science*, Okayama, Japan, Jun. 2016, pp. 1–7. doi: 10.1109/ICIS.2016.7550839.

[41] S. Niebuhr, K. Kohler, and C. Graf, "Engaging Patterns: Challenges and Means Shown by an Example," in *Engineering Interactive Systems*, J. Gulliksen, M. B. Harning, P. Palanque, G. C. van der Veer, and J. Wesson, Eds., in Lecture Notes in Computer Science, vol. 4940. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 586–600. doi: 10.1007/978-3-540-92698-6_35.