Al-Kharaj: Journal of Islamic Economic and Business
Volume 8 (1), 2026
P-ISSN: 2686-262X; E-ISSN: 2685-9300

A Modular Algorithmic Framework for Islamic Inheritance: Integrating Formal Fiqh Principles with Edge Case Calculation

Irfanun Nisa Tsalis Hantanty[1], Any Meilani[2], Muhammad Anis[3], Tirto Prayitno[4]
[1,2,3,4]*Universitas Terbuka, Indonesia*
Email: nisa.prayitno@gmail.com, any@ecampus.ut.ac.id, muhammad.anis@ecampus.ut.ac.id, tirtaiu@gmail.com

Keywords:
fiqh mawaris, special case, modular algorithm, object-oriented programming.

Abstract

The digitization of Islamic inheritance calculation is dominated by applications focused on settling common cases (masail ammah). The majority of previous research has neither addressed nor provided solutions to specific cases of inheritance distribution. Thus, there is a challenge to develop an inheritance calculation algorithm that addresses exceptional cases within the community. This study develops a modular algorithm framework for the calculation of mawaris fiqh that includes classical mechanisms (harmony, conditions, hajb, distribution of asbabul furud and 'asobah, awl, radd, and interaction of jaddul ikhwah) so that it is not only an application of the inheritance calculator and is accompanied by a transparent pseudocode. The approach taken is normative and translates the rules of mawaris fiqh, which includes exceptional cases, into modular algorithms and object-oriented programs. The algorithm's results are then implemented and tested to ensure consistency and accuracy of the calculations. The system has been proven to be able to handle several special cases, including Umariyyatayn, awl cases, radd without 'asobah, and jaddul ikhwah complex interactions. This study contributes to the standardization of algorithms for mawaris fiqh, including classical mechanism rules with exceptional cases, an explicit modular architecture based on fiqh stages, and formal pseudocode to ensure future maintenance. This framework provides a transparent alternative to the black-box LLM trend and can be the foundation for an Islamic inheritance system algorithm that can be examined and improved.

INTRODUCTION

Improvements to the inheritance algorithm in digital computation need to continue, given the complexity of problems and the evolving nature of society. The calculation of inheritance according to the main rules of the Qur'an remains the primary basis, in addition to being a standard rule in Islam, to avoid conflicts in the family that can have many effects. The inheritance calculation system requires a process of classification of heirs, a blocking mechanism (hajb), a fixed share distribution (ashhab al-furud), residual distribution (asabah), and proportional correction (awl and radd) that must be executed sequentially with precision (Bench-Capon & Sartor, 2003; Prakken & Sartor, 2015). Furthermore, it should not be forgotten that the provision of calculations for exceptional cases (masa'il syawadz) such as Umariyyatayn, al-akdariyyah, and jaddul ikhwah interactions requires multi-level reasoning (Hoque et al., 2022). The digitization of the Islamic inheritance system has been the focus of research since the last decade, with various approaches ranging from rule-based expert systems (Liao, 2005; Prentzas & Hatzilygeroudis, 2007), mobile applications (Md Zin et al., 2025), to Large Language Models-based approaches (AL-Smadi, 2025; AlDahoul & Zaki, 2025). However, a systematic evaluation by Billah (2023) of 10 Android applications shows that the majority of existing systems are not yet able to handle exceptional cases, with inconsistent outputs and a limited beneficiary list. These findings indicate a gap in the

standardization of classical fiqh algorithms for handling exceptional cases.

From a software engineering perspective, legacy systems typically use a monolithic architecture, which makes it difficult to verify, maintain, and add features (Garlan & Shaw, 1993; Kruchten, 1995). Hoque et al. (2022) propose a structured architecture using Data Flow Diagrams (DFD) and structure charts, but do not present explicit pseudocodes for specific mechanisms such as hajb, awl, and radd. Meanwhile, the latest trend is leading to a black-box LLM approach (AL-Smadi, 2025; AlDahoul & Zaki, 2025), which, despite achieving high accuracy, lacks algorithmic transparency and is difficult for fiqh scholars to verify manually.

Transparency and verifiability are important requirements in legal systems, including Islamic inheritance (Governatori et al., 2018; Palmirani et al., 2020). Expert systems for the legal domain require explainability that allows stakeholders to understand the reasoning process (Kumar et al., 2023; Alharbi & Alshammari, 2023). In the context of mawaris fiqh, this means that any division decision must be traceable to the underlying rules of fiqh, something challenging to achieve with a black-box approach.

Research Gaps

Based on an analysis of the literature on the digital Islamic inheritance system over the last five years, several fundamental methodological gaps were identified. So far, documentation of the steps of formal programming logic or pseudocode remains very limited, as most studies present only flowcharts or general process descriptions that are difficult for other researchers to reproduce. In addition, there has not been a single integrated framework that comprehensively documents all the mechanisms of classical fiqh, from the obstruction of heirs and the adjustment of inheritance through the aul and radd mechanisms to the complex calculations of the interaction between a grandfather and a brother. Current research tends to focus on only a few rules and has not yet implemented a modular architecture that divides the computational process into independent stages. The latest trend in the use of large language models (LLMs) offers high accuracy. However, the approach is closed, so it is not possible to verify the actual logic that needs to be implemented in the inheritance legal system.

Research Objectives and Contributions

This research aims to develop a modular algorithmic framework for calculating fiqh mawaris that addresses the various weaknesses outlined above. The main contribution of this study is the development of algorithm standards that fully accommodate all inheritance mechanisms, including harmony, conditions, and exceptional cases, so that the results of this study are not merely an application of an ordinary inheritance calculator but a formal, logical documentation. By applying a modular architecture divided into five main phases, each calculation stage has apparent limitations that facilitate the system's verification and maintenance. In addition, this study presents an in-depth validation of nineteen special-case patterns, such as Umariyyatayn, as well as other complex calculation conditions that are often not handled by existing systems. By offering an open model, this study provides an alternative for scholars, heirs, and legal practitioners to carry out the correction process manually and transparently.

METHODS

The normative-reconstructive approach is used to extract fiqh rules from classical literature and convert them into specific algorithmic structures. This approach consists of three main stages: rule extraction from normative sources, logic formalization, and module-based architecture.

In the rule extraction approach, we use the following reference sources: Al-Mughni by Ibn Qudamah (Hanbali madhhab), Bidāyat al-Mujtahid by Ibn Rushd (comparative 4 madhhabs), and *Al-Fiqh al-Islami wa Adillatuhu* by Wahbah al-Zuhayli (contemporary). Then, several identifications were carried out as legal domains (Palmirani et al., 2020; Governatori et al., 2018), among others, that identify normative propositions, namely that every fiqh rule is expressed in the form of an IF-THEN. Then, the classification of rules by fiqh stages, such as validation rules, hajb (blocking rules), distribution (allocation rules), and awl/radd (correction rules). Finally, the identification of the khilafiah was carried out in accordance with the rules governing cases of disagreement among the madhhabs. This study adopts the Shafi'i madhhab's opinion as a *baseline*.

The rules of fiqh that have been extracted are formalized using first-order logic and propositional logic (Hoare, 1969; Prakken & Sartor, 2015), e.g., "The father gets 1/6 if there are children (male or female), and gets residual as '*asobah* if there are no children." Created the logical formula as follows:

```
1    LET father ∈ Heirs
2    LET has_child = (∃ h ∈ Heirs : h.relation = "son" ∨ h.relation = "daughter")
3    IF has_child THEN
4    father.fardh_share = 1/6
5    father.is_asabah = FALSE
6    ELSE
7    father.fardh_share = 0
8    father.is_asabah = TRUE
9    END IF
```

This formalization allows for direct conversion to pseudocode with non-double-standard semantics.

The next approach is to divide architecture into several modules based on the stages of fiqh (Patel et al., 2021; Taibi et al., 2023). The division of modules follows the principle *of Separation of Concerns*, namely each module handles one stage of fiqh without internal interdependence (Parnas, 1972); *Interface-Based Design* where modules communicate through immutable *InheritanceCase* data structures, ensuring no side effects (Zdun & Avgeriou, 2005); and *Sequential Execution where* modules are executed sequentially without nested loops, achieving O(n) complexity (Zhou & Feng, 2023).

The system architecture is divided into 5 modules that are executed sequentially, namely the module validation of harmony and conditions, the hijab module, the fardhu module (Ashḥāb al-Furuḍ distribution), the Asabah module (the distribution of '*Asobah* + *Jaddul ikhwah*), and the Correction module (awl & Radd). To bridge the modular architecture with an operational view of computation, the overall process is summarized as a six-step scenario. Figure 1 presents the end-to-end flow from case input and validation through hijab filtering, fardh and ‘asabah allocation, and awl/radd correction, which serves as the baseline for the subsequent module specification and testing design.

1. The framework reads and validates the heir's data, the components of the tirkah (property), and the list of living heirs.
2. The framework applies a hijab module to identify precluded heirs and compile a list of valid heirs.
3. The framework assigns the fardh portion to all eligible ashab al-furudh and calculates the total fardh portion.
4. The framework distributes the remainder to the eligible 'asabah' heirs (including handling grandparent-sibling interactions if necessary).
5. The framework implements a correction mechanism by applying awl when the total fardh is >1 and applying radd when the total fardh is <1 and there are no 'asabah' (using makhraj/taṣḥīḥ if necessary).
6. The framework displays the final results of the inheritance distribution for each heir (proportion and/or nominal amount) along with the decision stages.

Figure 1. Six-step computation scenario of five modul Inheritance Algorithmic Framework

Based on the scenario in Figure 1, the five modules are specified in sequence as follows. The validation module receives input in the form of a list of heirs, inheritance data, and inheritance, and then issues a valid/invalid status along with an error message. The function of this module is to verify the completeness of the harmony and the validity of the inheritance requirements, including the certainty of the heir's death, the existence of the heir at the time of death, and the absence of māniʿ (inheritance barrier).

No	Heirs	Preconditions (as stated)	Computational Model (share of E)	Notes
1	Husband	Always if husband exists	E × 1/2 if NOT has_child; E × 1/4 if has_child	Furūḍ
2	Wife	Always if wife exists	E × 1/4 if NOT has_child; E × 1/8 if has_child	Furūḍ
3	Mother	Always if mother exists	E × 1/3 if NOT has_child and sib_count < 2; otherwise E × 1/6	Furūḍ (Shafiʿi: siblings must be ≥2 to reduce to 1/6)
4	Father (fardh part)	If father exists and has_child	E × 1/6	May later act as sharer/residuary (see notes)
5	Daughters (no son)	daughter exists and son does NOT exist	E × 1/2 if 1 daughter; E × 2/3 if ≥2 daughters	Furūḍ only when there is no son
6	Uterine sibling (single)	uterine_brother/uterine_sister exists, NOT has_child, father does NOT exist, and uterine count = 1	E × 1/6	Furūḍ
7	Uterine siblings (≥2)	Same as No. 6, uterine count ≥2	E × 1/3 shared per head	Furūḍ

8	Full sister	(1) No father, no children/grandchildren, and no full brother → furūḍ (1/2 or 2/3). (2) No father, no son/male grandchild, and no full brother, but a daughter/female grandchild exists → ʻasabah maʻa al-ghair.	E × 1/2 OR E × 2/3 OR ʻasabah maʻa al-ghair	Most frequently misapplied (Az-Zuhaili)

Table 1. Summary of Prescribed Shares

After the data is declared valid, the hijab module continues processing the list of valid heirs to produce a filtered list by eliminating those blocked under the priority rules. Next, the Fardhu Module uses a list of post-hajj heirs to determine the fixed share (asbabul furudh), namely 1/2, 1/4, 1/8, 2/3, 1/3, and 1/6, based on the classification of the heirs and the accompanying conditions. The remaining undivided property is then processed in the Asabah Module, which distributes the residue to the '*asobah* group, including special handling for jadd-ikhwah cases, with O(n) complexity. Finally, the Correction Module corrects when the total part is not equal to 1, by normalizing the part proportionally in the event of awl or redistributing in the event of radd.
Each module is validated using 19 testing units to ensure the accuracy of results at all stages of calculation. In the validation module, there are 3 tests, including verification of harmony (complete/incomplete), and māniʿ (inheritance barrier) conditions, such as murder and religious differences. The hijab module was tested with 3 scenarios: the child blocking the grandchildren, the father blocking the grandfather, and the siblings obstructing each other. In the fardhu module, 5 tests were conducted, covering cases of husbands with or without children, mothers with or without children/siblings, and single and plural daughters. Furthermore, the asabah module includes 4 tests, namely fathers as asabah, sons as residual recipients, and two jadd or ikhwah scenarios. Finally, the correction module was tested across 6 correction scenarios: awl with makhraj 27, awl with makhraj 15, radd without asabah, and radd with pairs.

3.4.2 Integration Testing

6 complex scenarios are used for integration testing:
Fardhu + Normal Asabah: Husband, mother, 2 daughters, son
awl Case: Husband, mother, 2 siblings (total = 1 + 1/6 + 2/3 = 15/12 → 15th awl)
Radd Case: Wife, mother, 2 daughters (total = 1/8 + 1/6 + 2/3 = 23/24 → radd remaining 1/24)
Hijab Complex: Father, grandfather, son, grandson (grandfather and grandson are hindered)
***Jaddul ikhwah*:** Grandfather, sibling, mother (grandfather chooses muqasamah or 1/3 of the remainder)
Umariyyatayn: Husband/wife, mother, father (mother gets 1/3 of the remainder, not 1/3 of the total)

3.4.3 Edge Case Master List

19 Patterns of Edge Cases Systematically Validated:
Category Hijab Complex:
Multiple blocking layers (anak → cucu → cicit)
Conditional blocking (siblings are blocked if you have children)
Partial blocking (father does not entirely block grandfather in distribution)

High awl category: 4. awl makhraj 27 (extreme cases) 5. awl makhraj 17 6. awl makhraj 15 (general)
Radd Category: 7. Radd without asabah (leftover to *asbabul furudh*) 8. Radd with partner (partner cannot radd) 9. Radd with Baitul Mal (no eligible heirs)
Jaddul Ikhwah *Category*: 10. Grandfather vs. sibling (muqasamah) 11. Grandpa vs. Brother 12. Grandpa chooses 1/6 (more profitable) 13. Grandpa chose 1/3 of the leftovers.
Umariyyatayn Category: 14. The first Umariyyatayn (husband, mother, father) 15. Second Umariyyatayn (wife, mother, father)
Ashabah Ma'al Ghayr Category: 16. Daughter + Son (asabah ma'al ghayr) 17. Sister + brother (asabah ma'al ghayr) 18. Granddaughter + grandson (asabah ma'al ghayr)
Combination Category: 19. awl + hijab + *jaddul ikhwah* (maximum complexity)

RESULTS AND DISCUSSION

Unit Testing Results

The development of algorithms indirectly results in a stable system. In the early stages of implementation, unit testing revealed several failures that occurred repeatedly. Overall, the algorithm experienced 7 test failures across iterations before reaching a consistent logical structure. These failures mainly occur in the correction mechanism ('awl and radd), the interaction between the jadd and the siblings (jaddul-ikhwah), as well as the phase execution sequence that is not entirely separated between the normal distribution and the special case. In some early experiments, the system produced a distribution that was mathematically correct but did not fully align with the more favorable choice structure in fiqh, particularly in evaluating the grandfather option.

Improvements are carried out through architectural restructuring into five modular phases that are executed sequentially: validation, hijab, fardhu, asabah, and correction. In addition, the special-case detection mechanism is explicitly separated from the normal distribution flow to avoid logical conflicts. After the fix iteration, the retest showed that all test units ran without errors.

In the final stage of development, 19 test units were successfully carried out, meeting expectations based on the calculation results. The test comprises five main modules: validation, hijab, fardhu, asabah, and correction. The entire module shows consistency of results after the logic structure is fixed.

To maintain mathematical precision, all calculations use rational arithmetic with fractions. fraction, so that no error occurs due to floating-point rounding. Validation is carried out by comparing the algorithm's output with manual calculations based on the reference work on farā'iḍ. After the refinement iteration, no deviation was found between the system results and the manual calculations.

Scientific Contributions

This study seeks to develop a formally documented algorithmic framework for calculating farā'iḍ by integrating the main mechanisms of classical fiqh mawaris into a single structure. So far, some previous studies have discussed several mechanisms, such as hajj, fardhu, and asabah, separately, or implemented inheritance calculators without detailed logical documentation. In this study, all the main mechanisms—including hajj, fardhu, asabah, makhraj, taṣḥīḥ, 'awl, radd, and jadd's interaction with brothers—are formulated in the form of an explicit and reproducible formal pseudocode. This framework was compiled not to replace classical jurisprudence literature, but to formalize its logical structure so that it can be re-implemented without ambiguity by other researchers. Thus, the contribution of this research lies in the effort to standardize algorithmic representations that have not previously been thoroughly documented within a single integrated framework.

In addition, this study designed an algorithmic architecture in five phases that follow the conceptual stages in fiqh mawaris: validation, hijab, fardhu, asabah, and correction. This separation is done explicitly to maintain clarity of the responsibilities of each module and minimize logic

overlap. The modular approach allows each phase to be tested independently through unit tests, enabling errors to be traced more precisely. In the early stages of development, the algorithm experienced several failures, especially in its correction and evaluation mechanisms for joint choices during interactions with siblings. However, by restructuring the flow and separating the privileged cases from the normal distribution, the system finally achieved stability. This modular structure also allows for further development, such as adding additional schools, without dismantling the entire system.

This study also compiles a classification of systematically tested case patterns. A total of nineteen patterns were designed based on the combination of mechanisms involved, such as complex hajb, 'awl, radd, jaddul-ikhwah, and combinations of several mechanisms at once. Each pattern is validated through formal testing and compared with manual calculations based on fiqh references. This approach results in a structured validation framework that other researchers can reuse to test similar systems. Thus, this research contributes not only to the implementation of algorithms but also to the provision of documented test patterns.

In the midst of the development of a large language model-based approach that is black-box, this study takes a white-box approach that emphasizes logical traceability. Every distribution decision can be traced back to the underlying rules of fiqh, so that the reasoning process is not hidden behind a statistical model. This transparency is considered important in the Islamic legal system, as it allows scholars, legal practitioners, and related parties to examine and, if necessary, manually understand the calculation process. With this approach, the research seeks to maintain a balance between computational precision and normative accountability.

Overall, the contribution of this research does not lie in the claim of absolute superiority, but in the preparation of algorithmic structures that are documented, modular, testable, and traceable back to the source of fiqh. This approach is expected to be a more stable basis for the development of a technology-based Islamic inheritance system in future research. The pseudocode in this study can be accessed at https://github.com/nisaprayitno-oss/IslamicInheritance/tree/main

CONCLUSION

This research produced a modular algorithmic framework for the calculation of mawaris fiqh, formulated as formal pseudocode and implemented systematically. This framework includes the main mechanisms in classical mawaris fiqh, ranging from the pillars and conditions, hajj hirman, the distribution of asbabul furudh, the distribution of 'asabah, including the interaction of jaddul ikhwah, to correction mechanisms such as 'awl, radd, makhraj, and taṣḥīḥ. The preparation of this framework departs from the need to formalize the structure of mawaris logic into a form that can be tested, reproduced, and traced back to the underlying rules of fiqh.

Conceptually, this study seeks to fill a gap in the literature by documenting all inheritance mechanisms within a single integrated algorithmic structure. Some previous studies have developed inheritance calculators or addressed specific aspects of distribution, but have not yet presented formal documentation that unifies all stages into a single, explicit, and modular architecture. In this study, the algorithm is organized into five phases that follow the conceptual flow of fiqh, namely validation, hajj, fardh, asabah, and correction. This separation is intended to maintain clarity about each phase's responsibilities while enabling further testing and development without altering the overall structure.

Documentation in the form of formal pseudocode is compiled so that it can be re-implemented by other researchers without dependence on one particular programming language. With this approach, reproducibility becomes more open, as algorithmic logic is embedded not only in the code but also conceptually described. The test results show that the developed framework can handle all the tested cases, including those that require special handling. More than that, the transparency of the algorithm allows the reasoning process to be traced manually by those with fiqh competence, so that the system does not stand as a closed mechanism.

Thus, the contribution of this research is not only to technical implementation, but to the preparation of algorithmic structures that are documented, modular, and verifiable. This framework is expected to serve as the basis for developing a more transparent and accountable Islamic inheritance system in the future.

Research Limitations

This research still has several limitations that need to be noted in proportion. The framework developed uses the Shafi'i madhhab as the primary basis for formulating rules and determining distribution priorities. Differences of opinion with other schools have been identified at the conceptual level, but they have not been fully implemented in operational configurations. Therefore, the development of other madhhabs requires more in-depth follow-up research.

In addition, this framework focuses on the distribution of inheritances after deducting wills and debts. The validation mechanism and the detailed calculations related to wills and debts have not been fully integrated into the system. Similarly, the distribution for dhawū al-arḥām has not been part of the current implementation, as the focus of the research is directed on the basic structure involving asbabul furudh and 'asabah.

From an implementation perspective, this study provides documented algorithms and pseudocode but lacks a user interface designed for the general public. Thus, direct utilization by non-technical users still requires advanced application development.

Advanced Research Suggestions

Given these limitations, several directions for future research can be considered. The development of multi-madhhab support is an important agenda, with a configuration approach that accommodates differences of opinion without overhauling the algorithm's core structure. The integration of the will and debt mechanisms is also needed to ensure the system more fully reflects the distribution process.

The development of a specific module for dhawū al-arḥām could be the next expansion, taking into account the approach appropriate to the madhhab in use. From an implementation perspective, providing web-based interfaces or mobile applications will expand access for non-technical users, while visualizing the reasoning process will maintain transparency.

At the methodological level, advanced research can employ formal verification to ensure greater mathematical rigor. Comparative studies between madhhabs in cases of khilafiah can also make a broader academic contribution. In addition, integration with larger Islamic legal technology ecosystems, such as waqf management or zakat calculation, can expand the practical benefits of this framework.

Overall, the modular algorithm framework developed in this study provides an initial foundation that is still open to further development. The principles of modularity and separation of responsibilities are expected to facilitate development without requiring sweeping changes to the system's basic structure.

REFERENCE

AlDahoul, N., & Zaki, Y. (2025). Benchmarking the legal reasoning of LLMs in Arabic Islamic inheritance cases. arXiv preprint arXiv:2508.15796. https://doi.org/10.48550/arXiv.2508.15796

Alharbi, A., & Alshammari, M. (2023). Explainable AI for Islamic finance decision support systems. Journal of King Saud University - Computer and Information Sciences, 35(7), Article 101890. https://doi.org/10.1016/j.jksuci.2023.101890

AL-Smadi, M. (2025). QU-NLP at QIAS 2025 shared task: A two-phase LLM fine-tuning and retrieval-augmented generation approach for Islamic inheritance reasoning. arXiv preprint arXiv:2508.15854. https://doi.org/10.48550/arXiv.2508.15854

Alshuqayran, N., Ali, N., & Evans, R. (2022). A systematic mapping study of microservice architecture. Journal of Systems and Software, 194, Article 111456. https://doi.org/10.1016/j.jss.2022.111456

Az-Zuhaili, W. (1997). Al-Fiqh al-Islami wa Adillatuhu. Dar al-Fikr.

Bench-Capon, T., & Sartor, G. (2003). A model of legal reasoning with cases incorporating theories and values. Artificial Intelligence, 150(1–2), 97–143. https://doi.org/10.1016/S0004-3702(03)00108-5

Billah, M. (2023). COMPLETE AND INCOMPLETE CALCULATION: Expert systems apps on the special cases of Islamic inheritance law. Al-Ahwal: Jurnal Hukum Keluarga Islam, 16(2), 201–216. https://doi.org/10.14421/ahwal.2023.16201

Bogner, J., Zimmermann, A., & Wagner, S. (2022). Automatically measuring the maintainability of service- and microservice-based systems. Journal of Systems and Software, 191, Article 111345. https://doi.org/10.1016/j.jss.2022.111345

Chen, L. (2022). Microservices: Architecting for continuous delivery and DevOps. IEEE Software, 39(2), 84–90. https://doi.org/10.1109/MS.2021.3128806

Di Francesco, P., Malavolta, I., & Lago, P. (2017). Research on architecting microservices: Trends, focus, and potential for industrial adoption. IEEE International Conference on Software Architecture (ICSA), 21–30. https://doi.org/10.1109/ICSA.2017.24

Dijkstra, E. W. (1968). Go to statement considered harmful. Communications of the ACM, 11(3), 147–148. https://doi.org/10.1145/362929.362947

Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2022). Learning from imbalanced data sets: Recent advances and challenges. Expert Systems with Applications, 201, Article 117098. https://doi.org/10.1016/j.eswa.2022.117098

Forgy, C. L. (1982). Rete: A fast algorithm for the many pattern/many object pattern match problem. Artificial Intelligence, 19(1), 17–37. https://doi.org/10.1016/0004-3702(82)90020-0

Gamma, E., Helm, R., Johnson, R., Vlissides, J., & Booch, G. (1993). Design patterns: Abstraction and reuse of object-oriented design. ECOOP'93—Object-Oriented Programming, 406–431. https://doi.org/10.1007/3-540-47910-4_21

García-Martínez, R., Britos, P., & Rodríguez, D. (2022). Validation and verification frameworks for rule-based expert systems. Knowledge-Based Systems, 248, Article 108847. https://doi.org/10.1016/j.knosys.2022.108847

Garlan, D., & Shaw, M. (1993). An introduction to software architecture. Advances in Software Engineering and Knowledge Engineering, 1, 1–39. https://doi.org/10.1142/9789812798039_0001

Governatori, G., Idelberger, F., Milosevic, Z., Riveret, R., Sartor, G., & Xu, X. (2018). On legal contracts, imperative and declarative smart contracts, and blockchain systems. Artificial Intelligence and Law, 26(4), 377–409. https://doi.org/10.1007/s10506-018-9223-3

Gupta, P., & Gupta, N. (2022). Algorithm design paradigms: A comprehensive survey. ACM Computing Surveys, 54(3), Article 52. https://doi.org/10.1145/3448734

Hassan, M. K., Rabbani, M. R., & Ali, M. A. M. (2022). Challenges for Islamic finance and banking in the post-COVID era and the role of Fintech. Journal of Economic Cooperation and Development, 43(3), 93–116.

Hayes-Roth, F. (1985). Rule-based systems. Communications of the ACM, 28(9), 921–932. https://doi.org/10.1145/4284.4286

Hoare, C. A. R. (1969). An axiomatic basis for computer programming. Communications of the ACM, 12(10), 576–580. https://doi.org/10.1145/363235.363259

Hoque, A. H. M. S., Tabassum, S., Nazib, A., Mustafa, R., & Rahman, M. O. (2022). An effective software architecture for the Islamic inheritance system employing a structured paradigm. Jurnal Kejuruteraan, 34(6), 1271–1284. https://doi.org/10.17576/jkukm-2022-34(6)-12

Knoche, H., & Hasselbring, W. (2019). Drivers and barriers for microservice adoption: A survey among professionals in Germany. Enterprise Modelling and Information Systems Architectures, 14, 1–35. https://doi.org/10.18417/emisa.14.1

Kruchten, P. (1995). The 4+1 view model of architecture. IEEE Software, 12(6), 42–50. https://doi.org/10.1109/52.469759

Kumar, A., Sharma, K., & Singh, H. (2023). Hybrid knowledge-based system for automated legal compliance checking. Knowledge-Based Systems, 267, Article 110456. https://doi.org/10.1016/j.knosys.2023.110456

Li, J., Wang, H. J., & Bai, X. (2023). A rule-based expert system with a bidirectional reasoning strategy. Expert Systems with Applications, 213, Article 118963. https://doi.org/10.1016/j.eswa.2022.118963

Li, Z., Liang, P., & Avgeriou, P. (2023). Understanding architectural technical debt: A systematic mapping study. Journal of Systems and Software, 198, Article 111589. https://doi.org/10.1016/j.jss.2023.111589

Liao, S. H. (2005). Expert system methodologies and applications—A decade review from 1995 to 2004. Expert Systems with Applications, 28(1), 93–103. https://doi.org/10.1016/j.eswa.2004.08.003

Md Zin, I. H., Mansor, N. F., Mat Diah, N., Hashim, S., & Mansor, M. I. (2025). Faraid distribution calculation using an AI-based Quranic chatbot. International Journal of Robotics and Automation, 14(3), 393–406. https://doi.org/10.11591/ijra.v14i3.pp393-406

Mehdiyev, N., Enke, D., Fettke, P., & Loos, P. (2022). Evaluating forecasting performance by decomposing market efficiency into three components. Expert Systems with Applications, 203, Article 117449. https://doi.org/10.1016/j.eswa.2022.117449

Pahl, C., & Jamshidi, P. (2021). Microservices architecture: A systematic mapping study. Journal of Systems and Software, 182, Article 110987. https://doi.org/10.1016/j.jss.2021.110987

Palmirani, M., Martoni, M., Rossi, A., Bartolini, C., & Robaldo, L. (2020). Legal ontology for modelling GDPR concepts and norms. Artificial Intelligence and Law, 28(2), 149–169. https://doi.org/10.1007/s10506-020-09267-8

Parnas, D. L. (1972). On the criteria to be used in decomposing systems into modules. Communications of the ACM, 15(12), 1053–1058. https://doi.org/10.1145/361598.361623

Patel, M., Virparia, P., & Patel, D. (2021). A modular architecture for intelligent decision support systems. Knowledge-Based Systems, 231, Article 107432. https://doi.org/10.1016/j.knosys.2021.107432

Prakken, H., & Sartor, G. (2015). Law and logic: A review from an argumentation perspective. Artificial Intelligence, 227, 214–245. https://doi.org/10.1016/j.artint.2015.06.005

Prentzas, J., & Hatzilygeroudis, I. (2007). Categorizing approaches combining rule-based and case-based reasoning. Expert Systems, 24(2), 97–122. https://doi.org/10.1111/j.1468-0394.2007.00423.x

Shortliffe, E. H., & Buchanan, B. G. (1975). A model of inexact reasoning in medicine. Mathematical Biosciences, 23(3–4), 351–379. https://doi.org/10.1016/0025-5564(75)90047-4

Singh, R., & Gill, N. S. (2022). Ontology-based knowledge representation and reasoning in expert systems. Knowledge-Based Systems, 254, Article 109630. https://doi.org/10.1016/j.knosys.2022.109630

Soldani, J., Tamburri, D. A., & Van Den Heuvel, W.-J. (2021). The pains and gains of microservices: A systematic grey literature review. Journal of Systems and Software, 175, Article 110915. https://doi.org/10.1016/j.jss.2021.110915

Soto, J., Melin, P., & Castillo, O. (2022). A modular neural network architecture with fuzzy response integration for complex time series prediction. Expert Systems with Applications, 186, Article 115705. https://doi.org/10.1016/j.eswa.2021.115705

Susskind, R. (1987). Expert systems in law: A jurisprudential approach to artificial intelligence and legal reasoning. The Modern Law Review, 50(2), 168–194. https://doi.org/10.1111/j.1468-2230.1987.tb02575.x

Taibi, D., Lenarduzzi, V., & Pahl, C. (2023). Architectural patterns for microservices: A systematic mapping study. Journal of Systems and Software, 203, Article 111723. https://doi.org/10.1016/j.jss.2023.111723

Washizaki, H., Guéhéneuc, Y.-G., & Khomh, F. (2019). A taxonomy of software architecture patterns for distributed systems. IEEE Access, 7, 176888–176904. https://doi.org/10.1109/ACCESS.2019.2957887

Woodcock, J., Larsen, P. G., Bicarregui, J., & Fitzgerald, J. (2009). Formal methods: Practice and experience. ACM Computing Surveys, 41(4), Article 19. https://doi.org/10.1145/1592434.1592436

Zdun, U., & Avgeriou, P. (2005). Modeling architectural patterns using architectural primitives. ACM SIGPLAN Notices, 40(10), 133–146. https://doi.org/10.1145/1103845.1094824

Zhou, W., & Feng, L. (2023). Optimization strategies for large-scale rule-based inference engines. Knowledge-Based Systems, 272, Article 110589. https://doi.org/10.1016/j.knosys.2023.110589

Zimmermann, O. (2022). Architectural refactoring: A task-centric view. IEEE Software, 39(4), 77–84. https://doi.org/10.1109/MS.2022.3167936