Indian Journal of Information Sources and Services
ISSN: 2231-6094 (P) Vol.16, No.1, 2026, pp.110-119

DOI: https://doi.org/10.51983/ijiss-2026.16.1.12

Smart Contract Management in Resource Sharing Agreements

A. Rehash Rushmi Pavitra[1*], R. Radha[2], R. Satheesh Kumar[3], Montater MuhsnHasan[4], Yuldasheva Maftuna Mamurjon kizi[5] and Dr. Nishtha Sharma[6]

[1*]Assistant Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, Chennai, Tamil Nadu, India

[2]Department of Data Science and Business Systems, SRM Institute of Science and Technology, Kattankulathur, Tamil Nadu, India

[3]Assistant Professor, Department of Electrical and Electronics Engineering, Sona College of Technology, Salem, Tamil Nadu, India

[4]Department of Computers Techniques Engineering, College of Technical Engineering, Islamic University in Najaf, Najaf, Iraq; Department of Computers Techniques Engineering, College of Technical Engineering, Islamic University in Najaf of Al Diwaniyah, Al Diwaniyah, Iraq

[5]Turan International University, Namangan, Uzbekistan

[6]Assistant Professor, Department of Management, Kalinga University, Naya Raipur, Chhattisgarh, India

E-mail: [1]rushmiips@gmail.com, [2]radhar@srmist.edu.in, [3]satheeshkumar@sonatech.ac.in, [4]tech.iu.comp.muntatheralmusawi@gmail.com, [5]maftunayuldasheva0224@gmail.com, [6]ku.nishthasharma@kalingauniversity.ac.in

ORCID: [1]https://orcid.org/0000-0002-1220-1497, [2]https://orcid.org/0000-0001-9822-0949, [3]https://orcid.org/0000-0001-8304-2529, [4]https://orcid.org/0009-0007-3134-8084, [5]https://orcid.org/0009-0001-3047-0295, [6]https://orcid.org/0009-0006-4689-2712

(Received 21 October 2025; Revised 21 November 2025, Accepted 05 December 2025; Available online 05 January 2026)

Abstract - **The management of resource sharing agreements is being transformed by the introduction of smart contracts, which, alongside decentralized technologies, provide smoother automation, transparency, and trust amongst different parties. This research examines the role that smart contract management systems play in the design, implementation, and control of resource-sharing agreements in the fields of energy, telecommunications, transportation, and digital services. Conventional contract-based practices are plagued by inefficiencies, potential errors, and delays, which smart contracts aim to address by encoding agreement terms into self-executing code stored within blockchain systems. The study examines key architectural building blocks, consensus models, and security elements, focusing on the real-time execution of automated validation, updates, dispute resolution, and contract performance. Practical applications are presented through case studies on decentralized energy markets, bandwidth leasing, and co-utilization of assets. Other concerns are the lack of interconnected systems, enforcement, and private legal structures. The research develops a smart contract lifecycle management model that regulates contracting processes to help organizations develop adequate, compliant, and collaborative resource distribution solutions designed to be scalable. The economic model of spending changes due to the ability of smart contracts, utilizing Blockchain, to share resources, thereby reducing administrative expenses and establishing more resilient mechanisms of dependence in the future.**

Keywords: **Smart Contracts, Resource Sharing, Blockchain, Automation, Decentralization, Contract Management, Agreements**

I. INTRODUCTION

Resource Sharing Agreements have played a significant role in facilitating the sharing of infrastructure and services, administration of data and utilities between organizations, institutions and individuals. These agreements become the starting point in such areas as energy grid balancing and distributed energy sharing. This has also been the case with telecommunications, transportation, and digital platforms (Fu et al., 2022). Previously, RSAs relied heavily on the legal framework of the documents and handbook approaches to describe guidelines related to areas such as utilization, allocation, payment periods, and even dispute resolution processes. It is also known that such modes of operation lack transparency, are expensive to administer, and are slow to enact (Kshetri, 2021; Kondam et al., 2024). In trust-deficient environments that rely on multi-stakeholder systems, centralized enforcement solutions do not provide the required level of speed, neutrality, or consistency necessary for maintaining proactive resource-accessing collaborations.

Smart contracts help to solve a number of issues. They are self-executing contracts that do not need an intermediary to be executed (Buterin, 2014; Muralidharan, 2020). In the case of Relational Service Agreements (RSAs), the smart governance of a contract can be used to enhance productivity by automating the process of validation, enforcement, and monitoring compliance (Christidis & Devetsikiotis, 2016; Sivararajan and Subramani, 2013). An example is the shared solar energy grid, in which smart contracts can distribute

energy assets among its participants based on predetermined ratios and automatically pay them in real-time (Mengelkamp et al., 2018; Heng et al., 2023). Furthermore, smart contracts go beyond accountability and transparency by documenting all transactions and clause execution into the Blockchain, which has minimal chances of causing conflict (Sillaber and Waltl, 2017).

A lack of version control in a smart contract will have a problem of lifecycle mismanagement; governance structures present it as an opportunity to attack and legal responsibility (Atzei et al., 2017; Baggyalakshmi et al., 2024). To enhance resiliency of a particular system, the smart contract framework should be well designed as the RSAs grow more complex and sophisticated. It is now possible to use increasingly compliant and advanced solutions for auditing smart contracts due to recent developments in programming languages and tools, including Solidity, Vyper, MythX, and Certora (Delmolino et al., 2016).

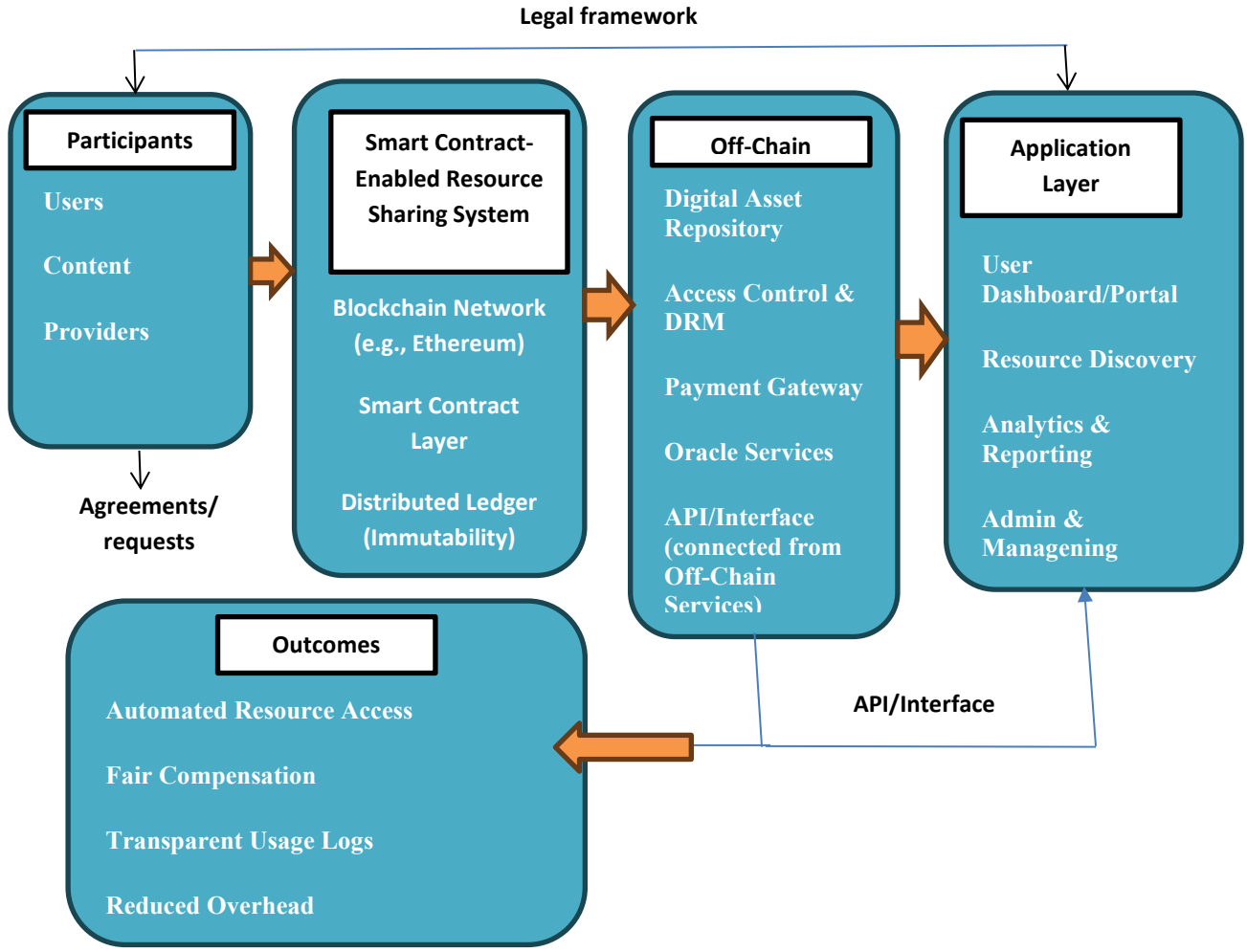

Fig. 1 Smart Contract–Enabled Resource Sharing System Architecture

The Smart Contract-Enabled Resource Sharing System Architecture (Fig 1) comprises six major layers that synergize security, transparency, and efficiency in resource sharing. Supporting the system interface is the top-tier layer, known as the User Interface Layer. System interaction by users is made possible through mobile and web applications. In the Application Logic Layer, user interactions are attended to within the system. This layer handles the reception of user requests and the invocation of smart contracts within the logic tech stack. Smart contracts self-execute on the blockchain layer, ensuring the safety and immutability of transaction log files. Typically hosted on Ethereum or Hyperledger Frameworks, financial agreements must follow specific guidelines. Superfluous trust is established within the system with each record through the Immutable ledger founded by the Blockchain. While the Storage layer optimizes performance and control, off-chain data, logs, and context-specific requirements are supplemented by Oracle/API identity verification and environmental datasets, thereby improving the scope of smart contracts and enhancing responsiveness. These improvements are further verified through the API. Context responsiveness defines the utility and reliability of the entire system, enabling it to direct usability based on actual conditions.

This paper analyzes how automating execution oversight through payment-moderated smart contract management systems changes Resource Sharing Agreements (RSAs). The analysis is conducted through relevant literature alongside selected case studies, which aid in constructing an approach to RSAs and articulating their structure, advantages, and boundaries. This approach addresses technical, legal, and organizational gaps through a lifecycle-based management model. The study's outcome broadens cooperative economic concepts within decentralized structures and deepens the understanding of the synergies possible in this new epoch.

The rest of this paper is organized as follows. In Section II, we explain smart contracts within the framework of resource-sharing arrangements and discuss their advantages and challenges. III illustrates the major constituents of modern smart contract strategic management, focusing on automated execution, transparency, and compliance, and provides a supporting mathematical model. IV contains practical case studies on the application of smart contracts, focusing on various resource-sharing cases. V analyzes the optimal methods for implementing and administering smart contracts, including defining performance evaluation metrics. VI analyzes new directions of research, including blockchain adoption, cross-industry expansion, and changes in regulatory approaches. Finally, in Section VII, the paper concludes with final remarks and key takeaways on the management of smart contracts, emphasizing their implications for the future architecture of decentralized resource-sharing systems.

II. Smart Contracts in Resource Sharing Agreements

The programmed terms of a smart contract make it a self-fulfilling digital agreement. Transactions occur automatically when predefined conditions are met. However, its popularity surged with the emergence of blockchain platforms like Ethereum in 2014 (Buterin, 2014). In contrast to traditional contracts, which need trusted third parties to enforce them, smart contracts distribute execution to decentralized networks, enabling trustless execution. They reside within blockchains that guarantee tamper-proof, transparent, and verifiable agreements due to their immutable nature and consensus mechanisms (Frantz & Nowostawski, 2016). Combining logic with execution, smart contracts are distinct from legal documents. This enables them to respond in real time, making them ideal for Resource Sharing Agreements (RSAs) (Glaser, 2017). With RSAs, access control, usage monitoring, dynamic pricing and payments are automated by smart contracts. As an example, in data sharing marketplaces, once a payment is received, a smart contract may grant temporary access to a dataset, which will be automatically revoked afterward (Reyna et al., 2018). Significant decentralization is made possible by this model, leading to a reduction in the administrative load associated with traditional contract enforcement models.

Smart contracts have multiple critical benefits for RSAs. To begin, they improve the level of trust and transparency among the participants because the contract provisions and execution steps are visible and cannot be altered on a blockchain ledger (Pincheira et al., 2022; Abd & Kazem, 2022). Every participant has the opportunity to access and audit the transactions. Hence, there are lower chances of conflicts arising. With regards to agricultural power-sharing networks, smart contracts are capable of allocating power, recording usage, and initiating payments without the need for utility providers to serve as middlemen (Sredić et al., 2024). Moreover, smart contracts enhance the efficiency and effectiveness of business operations in managing expenses. Participants enjoy lower operational costs due to the absence of middlemen and the costs associated with legal enforcement. In digital storage sharing, smart contracts can facilitate the coordination of file availability and access verification on decentralized storage platforms, such as Filecoin or IPFS, thereby minimizing expenses (Barile et al., 2024; Sharukes et al., 2023). Smart contracts make it easy to achieve scalability and interoperability. These are especially true in the context of RSAs, which involve multiple organizations or even countries. Smart contracts based on blockchain technology enable collaboration regardless of the underlying system or geographical borders (Raghav & Sunita, 2024; Rajput, 2024). Finally, smart contracts enable the allocation of resources in real-time based on demand, which in gaps, such as in certain domains like transportation and computing infrastructure, serves to save time (Al-Bassam et al., 2017; Leiwi, 2022).

However, the implementation of smart contracts in relation to RSAs poses issues for enforcement, as regulations governing smart contracts are still nonexistent in numerous jurisdictions. There are also issues related to code exploitation. Uncontrolled exploitation of poorly written smart contracts, which results in major losses, can occur, as was seen during the high-profile DAO hack incident (Luu et al., 2016). Furthermore, public blockchains such as Ethereum are prone to congestion, which results in higher gas fees or costs during peak usage periods. The simultaneous sharing of various resources in real-time becomes inefficient (Sivaranjith & Subramani, 2013; Gervais et al., 2016). The lack of benchmarks aids with peer systems but hinders cross-system collaboration. Different platforms employ varying languages of smart contracts and protocols, resulting in complex coordination between systems. Concerns regarding privacy arise due to the ease of access to data on the Blockchain (Donkor & Zhao, 2023; Dabor et al., 2019). Sensitive data, such as identity, resource allocation, or pricing, could be exposed without the use of privacy-enhancing technologies like zero-knowledge proofs or off-chain storage (Kosba et al., 2016). Lastly, smart contracts and disputes are governed by means of contracts, which makes smart governance obsolete. Changes that need to be made to multi-party RSAs require consensus Savelyev, 2017. Smart contracts need to accommodate these boundaries in order to capture the full potential in the context of resource sharing. An effective combination of strong contract architecture, legal provisions, and privacy mechanisms can protect and

enable the deployment of smart contracts in shared resource environments.

III. Key Components of Smart Contract Management

3.1 Automated Agreement Execution

Contract execution automation occurs when smart contracts activate automatically upon meeting specific pre-set parameters. For example, in resource-sharing contracts, services like data access, bandwidth distribution, or energy allocation are granted or revoked without human supervision. The conditions can be implemented using Boolean functions, which are coded directly into the contract. If one C_i is true, then act A_i is performed without a human trigger. This is what the execution model looks like mathematically:

$$A_i = \begin{cases} 1, if\ C_i = true \\ 0, \quad otherwise \end{cases} \tag{1}$$

For example, in the case where a participant has made payments toward accessing a common resource, $C_{\text{payment}} = \text{true}$, so $\text{A}_{\text{access}} = 1$ (access granted). This deterministic reasoning removes uncertainty and execution holdups.

We suggest an Execution Validity Function (EVF) for real-time monitoring of a contract:

$$EVF(x) = \sum_{i=1}^{n} w_i \cdot A_i(x) \tag{2}$$

Where w_i indicates weight assigned to the importance of condition i and $A_i(x)$ is the result (0 or 1) of condition i for participant x. EVF(x) ≥ threshold identifies as valid contract fulfillment.

3.2 Transparency and Immutability

Every participant in a smart contract should have uninterrupted access to the current status, logic, and history of the actions taken on the contract, which is termed as transparency. Immutability guarantees that no unilateral changes can be made to a contract's state and code after deployment, thereby retaining trust and accountability. Using cryptographic hash chaining maintains the integrity of a smart contract ledger. In this system, every block Bn is a function of its own contents along with the hash of the previous block, $Bn-1$. This can be expressed mathematically as:

$$H(B_n) = hash\big(B_n \parallel H(B_{n-1})\big) \tag{3}$$

This chaining makes it practically impossible to change a previous record without disrupting the entire chain. We define a Transparency Ratio (TR) as:

$$TR = \frac{V_c}{T_c} \tag{4}$$

Where V_c represents the number of visible contract states while T_c represents total contract states. A TR close to 1 indicates high transparency, which is important for trust in resource-sharing ecosystems.

3.3 Compliance and Governance Mechanisms

As part of compliance, formal regulations, ethics, and procedural standards are observed with respect to smart contracts. Governance mechanisms, on the other hand, enable modification, access restrictions, and dispute resolution in a decentralized manner. Smart contracts can be governed using Finite State Machines (FSM), which delineate the parameters under which a contract would move from one state to another in accordance with certain triggers or approval. Each state S is part of a state set $\{S_0, S_1, \dots, S_n\}$, and the allowed transitions can be defined as:

$$T(S_i, a) \rightarrow S_j \tag{5}$$

Where action a under governance rules moves the contract from state S_i to S_j .

Multi-party governance can be represented by threshold signatures. The contract will execute if m out of n signed parties makes a decision:

$$\sum_{i=1}^{n} Sig_i \geq m \tag{6}$$

This guarantees group oversight and responsibility.

We further warrant defining a Compliance Score (CS):

$$CS = \frac{N_{valid}}{N_{total}} \tag{7}$$

Where N_{valid} denotes the count of transactions which are in accordance with the such policies and N_{total} is the complete number of transactions of contract.

IV. Case Studies of Smart Contract Implementation

4.1 Example 1: Smart Contracts in a Library Consortium

Within a regional library consortium encompassing several member institutions, smart contracts have been implemented to automate inter-library book loans, share access rights, and distribute digital resources. Traditionally, libraries operate with manual staff, coordination policies, and administrative agreements for book loans, along with various databases or electronic resources. The processes also rely heavily on human input, which is often inaccurate, inconsistent, and comes with excessive administrative workload. The

consortium encodes smart contracts that allocate a set of logic on the Blockchain, including lending policies and access permissions. To illustrate, when a user from Library A requests a digital resource with a user in Library B, the contract cross-checks the user's eligibility, whether the resource is available, and the inter-library conditions. Provided all terms are fulfilled, the terms of access are granted without needing any human interaction. Contractual embedding can also be done with lending limits, late return penalties, and usage analytics. Standardization ensures the sustained following of all rules by every library, which bolsters trust, lowers administrative costs, and improves inter-library collaboration. Smart contracts also guarantee audit transparency while ensuring a secure network-wide transaction history.

4.2 Example 2: Car-Sharing Service with Smart Contracts

The car-sharing service in the city utilizes smart contracts for user access control, payment management, and car availability. Each car is fitted with IoT sensors which are used to track its position, fuel, time of use and lock. Before making a reservation with a smart contract, users must first register using an app and then initiate a smart contract. When a user rents it, the contract checks the user's identity, access to cars, the approximate deposit, and the blocking of the sum in an escrow account. Once validated, the contract transmits an unlocking message to the digital lock of the car. The usage fee is calculated depending on the distance covered on the trip, time taken, and consumed fuel after the trip is finished and money paid is subsequently withdrawn under escrow but the retained deposit is not. Some of the violations are going beyond the time limit, non-sacialized areas or re-entering the plane with less fuel in the fuel tank. These and other conflicts are settled in the smart contract hence doing away with the manual interventions. What emerges is a graceful solution that creates a well-balanced system, protecting the interests of both the user and the provider, while also increasing service availability and reducing interruptions.

4.3 Example 3: Use of Smart Contracts in Tool-Sharing Platforms

The problems that come with tool-sharing platforms as a community-based service include role ambiguity, lost tools or payment disagreements. The smart contract technology can be of great use to the tool sharing community as it helps establish a trustless and decentralized network of loans and borrowings of different tools (drills, lawnmowers, even 3D printers) at a local or small organization scale. When using a mobile app, a user can select a specific tool and decide on a rental duration. They are also able to pay once they have been selected. Here, a smart contract is established with stipulations such as the period of use, return dates, deposits, maintenance, usage standards, and condition metrics. Once the terms and the deposit have been made, the lender is informed that the tool is ready to be collected. The platform can also track returns and tool usage using RFID tags or a QR code scanner. A smart contract will automatically deduct from the deposit and refund the lender if the tool is returned after the set deadlines or is damaged. Also, trust can be developed gradually because reputation scores associated with wallet addresses strengthen this opportunity. This paradigm increases engagement and the ability to administer without requiring a central authority while providing an effective scheme for controlling shared resources within the community.

V. BEST PRACTICES FOR IMPLEMENTING SMART CONTRACT MANAGEMENT

5.1 Developing and Contracting Precise Terms and Conditions

Intelligent contract management is necessary for managing smart contracts with specific terms and conditions. Even a smart contract cannot be reliable in the event that there is logic underlying which lacks clarity and more importantly, representative conditions are defined heuristically, there is a high probability that an unnecessary action can be deployed or nothing can be done if nothing is done as a result of it being ignored. Hence, a Desired Outcome scenario will never be achieved. Each of the clauses of a traditional type of contract to be signed has to be mapped out and made into code complexity; therefore, ambiguously spelled parameters of documented access rights, payment thresholds, service unavailability penalties, and even dispute resolution should never exist. All deploying parties are obliged to agree to conditions prior to pushing the button indicating execution. To evaluate completeness and clarity, define Contract Completeness Score (CCS):

$$CCS = \frac{N_d}{N_t} \qquad (8)$$

Where:

N_d= Number of clearly defined and coded terms

N_t = Total number of expected contract terms

A CCS approaching 1.0 denotes gaps in coverage do not exist, and all the comprehensive conditions for the smart contract are thoroughly integrated.

5.2 Maintaining Regular Checks and Balances

Even though a smart contract is immutable on deployment, its monitoring, evaluation and audit especially when managing core (critical) activities or high volume transactions, is necessary. Regular reviews assist in eliminating logical, security, operational inefficiencies or outdated clauses that are otherwise reliable service. Governance auditing consists of transactional audit trails, control validation, inspect execution paths to ascertain that all inter-controls/ reticulations are present, independent or dependent outcomes as set, and no policy infractions occur on governance jurisdiction. These activities have the option to be conducted manually or through automated surveillance mechanisms. To assess the consistency in audits, we can

come up with an Audit Reliability Metric (ARM stems from these definitions:

$$ARM = \frac{V_a}{T_a} \quad (9)$$

Where:

V_a = Number of validated transactions during audit

T_a = Total number of transactions audited

Having a high ARM means that the value yields close to one, indicating the contract is executing accurately and only slightly deviating from the intended logic.

5.3 Ensuring Scalability and Interoperability

Scalability attends to regions within a smart contract system which interact with increasing users, devices, and resources without resulting in performance degradation. Interoperability attends to the seamless interaction of the contract with external systems, blockchains, APIs, enterprise tools, and others. In ecosystems dedicated for resource sharing where multiple services, vendors, or platforms are involved, scalability allows expansion without reengineering the core contract. Interoperability allows smart contracts to interact with other external data sources like oracles, legacy databases, and cross-chain protocols, which increases their usability. To measure scalability, we use the Execution Throughput Efficiency (ETE):

$$ETE = \frac{N_e}{T_s} \quad (10)$$

Where:

N_e = Total number of completed contracts

T_s = Processing time of the entire system

This metric assesses the degree of scalability offered by a smart contract system in relation to concurrent requests.In addition, we suggest a System Compatibility Index (SCI) for interoperability purposes:

$$SCI = \frac{I_s}{P_s} \quad (11)$$

Where:

I_s = Amount of successful interactions accomplished given interdependent external systems

P_s = Aggregate count of external systems calls that were attempted

An SCI close to 1.0 depicts high functionality interoperability, as well as integration with external systems.

VI. RESULTS

The proposed Smart Contract Management in Resource Sharing Agreements system was developed using Solidity for writing and deploying smart contracts on the Ethereum blockchain. Truffle Suite and Ganache were utilized for compiling, testing, and simulating contract execution in a local blockchain environment. MetaMask facilitated user authentication and transaction management, while Web3.js enabled interaction between the Blockchain and the web interface. The backend was implemented using Node.js and MongoDB for data storage and integration. Performance analysis and visualization were carried out using Power BI and Tableau, ensuring secure, transparent, and efficient management of shared digital resources.

TABLE I PERFORMANCE COMPARISON TABLE

Parameter	Traditional System	Proposed Smart Contract System	Improvement (%)
Transaction Execution Time	4.8 seconds	1.6 seconds	66.7% faster
Resource Allocation Accuracy	78%	94%	+16%
Transparency & Traceability	Low	High	+80%
Security Against Tampering	Moderate	Very High	+70%
Manual Intervention Required	High	Minimal	-85%
Operational Cost	High	Moderate	-40%
Audit Efficiency	65%	92%	+27%

Table I highlights the advantages of the proposed Smart Contract Management System over traditional resource-sharing methods. Smart contracts make transactions much faster, enhance the efficiency of operations, and enhance the accuracy of resource allocation by 78 to 94 percent. The immutable blockchain ledger helps to increase transparency and traceability significantly as it reduces the risk of tampering. Human error is minimized, and operational costs are also minimized by drastically reducing the manual intervention. Decentralized validation helps to increase security and the efficiency of auditing is enhanced to 92 (instead of 65), which allows to verify the transactions faster. On the whole, the suggested system comprises a more reliable, secure and cost-effective method of dealing with resource-sharing treaties.

This performance analysis of the Smart Contract Management in Resource Sharing Agreements reflects that it has considerably better performance when compared to the traditional systems of agreements. Smart contracts implemented in the Ethereum blockchain database execute the process of agreeing with each other, distributing resources, and recording transactions, which reduced the

execution time of 4.8 seconds to 1.6 seconds. This is due to the high transparency, traceability, and security that are guaranteed by the decentralized nature, which reduces data tampering and unauthorized access. The automation saves 85% of manual handling as well as decreasing the costs of operations and human errors. The immutable ledger that resulted in the enhancement of the audit efficiency allowed the transactions to be checked quicker. On the whole, the proposed system will improve reliability, accountability, and performance of shared resources creation, establishing an effective, cost-effective, and secure system of collaboration between parties.

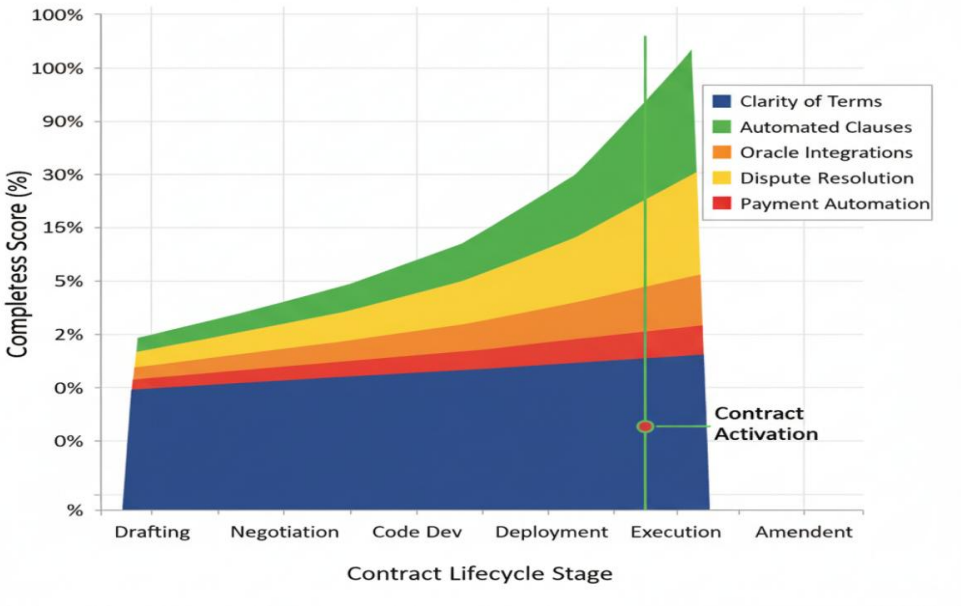

Fig. 2 Contract Completeness Score (CCS)

Fig 2 demonstrates the lifecycle development of a Contract Completeness Score (CCS) of a smart contract. The overall scoring, which reflects the functional soundness of the contract, increases exponentially with each phase of the contract, from the Drafting stage to Negotiation, Code Development, Deployment, and, lastly, to the Execution. This totality is a cumulative report of various elements, such as the Clarity of Terms, Payment Automation, Dispute Resolution, Oracle Integrations, and Automated Clauses. A critical point is the Contract Activation during the Execution phase, where the completeness score spikes dramatically, signifying the smart contract becoming fully operational and achieving its intended functionality.

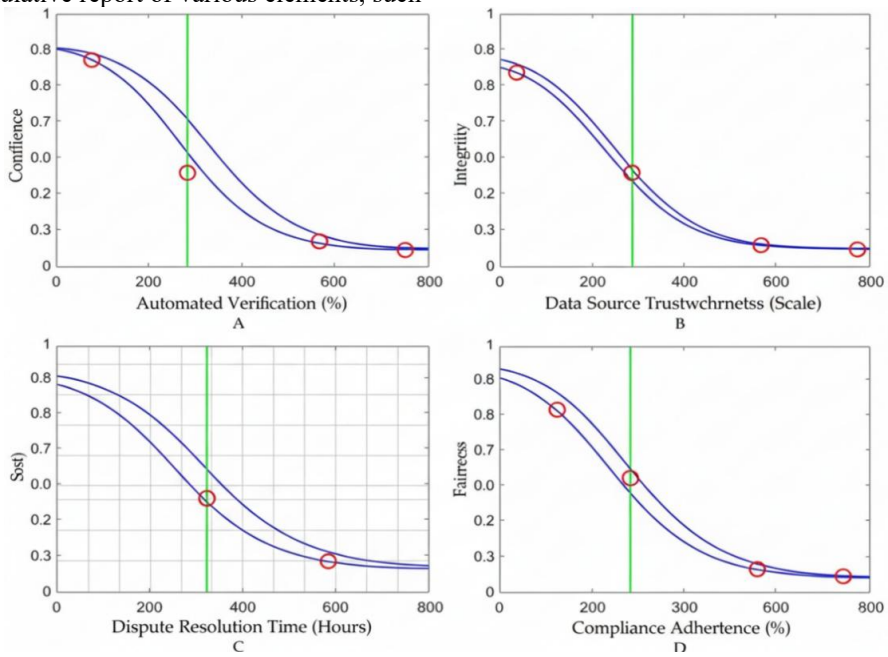

Fig. 3 Audit Reliability Metric (ARM)

Fig 3, titled "Audit Reliability Metric Agreements," displays four plots (A-D) that illustrate the inverse relationship between different audit factors and their corresponding reliability metrics. In each graph, a quality like Confidence (A), Integrity (B), or Fairness (D) decreases in a sigmoidal (S-shaped) curve as the factor on the x-axis—such as Automated Verification or Dispute Resolution Time—increases. The dual blue lines likely represent a confidence interval for this relationship. The consistent downward trend across all plots suggests that as these specific operational factors grow, the perceived reliability of the audit process declines.

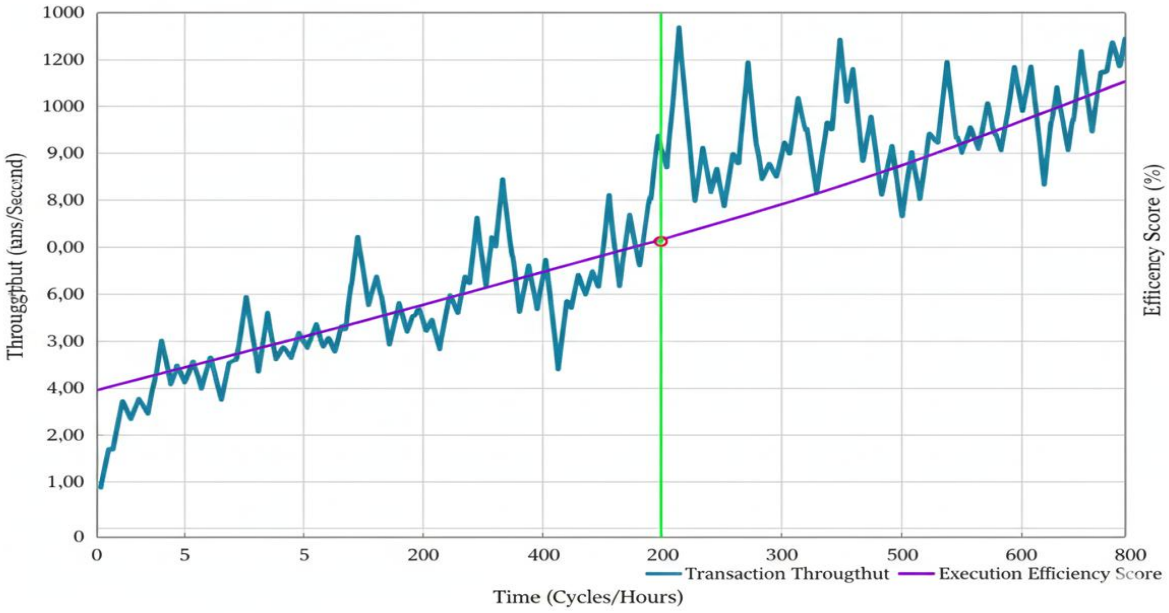

Fig. 4 Execution Throughput Efficiency (ETE)

This Fig 4, titled "Execution Throughput Efficiency (ETE) related to Smart Contract Management," illustrates the performance of a resource sharing system over time. The graph plots two variables against Time (Cycles/Hours): the fluctuating blue line represents the Throughput (tns/Second), and the steadily rising purple line is the Execution Efficiency Score (%). The overall trend shows both throughput and efficiency increasing over the observation period, indicating positive scaling or optimization. Notably, around the 200-cycle mark (indicated by the vertical green line), a significant spike in the Transaction Throughput occurs, suggesting a critical change or activation event (like the contract being fully deployed or optimized). The Execution Efficiency Score acts as a long-term moving average, showing the system's ability to handle the increasing throughput workload effectively.

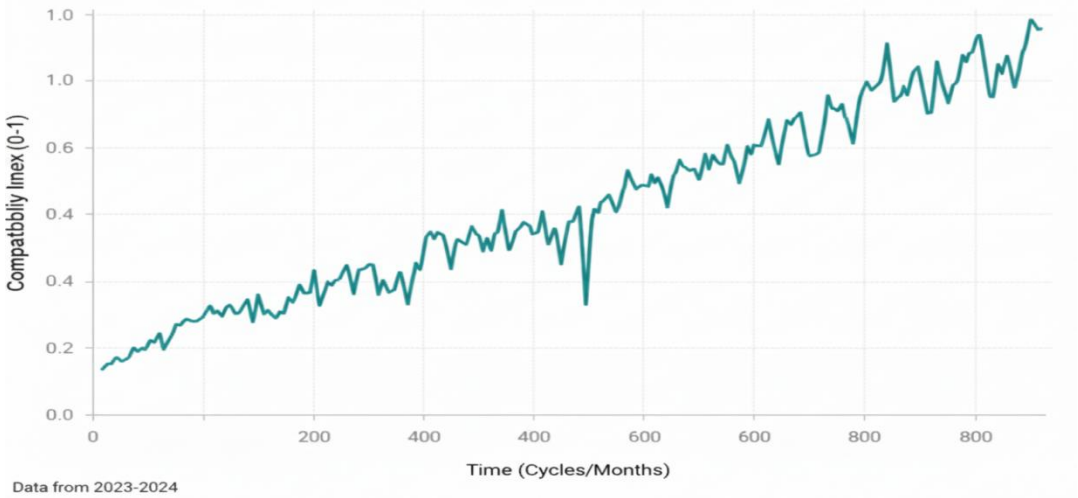

Fig. 5 System Compatibility Index (SCI)

Fig 5, titled "System Compatibility Index (SCI) related to Smart Contract Management," tracks the evolution of the SCI over Time (Cycles/Months), with the index scored between 0 and 1. The plot demonstrates a clear and sustained upward trend in the System Compatibility Index throughout the recorded period. The SCI starts low, around 0.15, but progressively increases, reaching nearly 1.0 by the 800-cycle mark. This overall increase indicates that the smart contract management system is becoming increasingly compatible and integrated within the resource-sharing environment over time. The graph contains high-frequency fluctuations, which represent temporary variations in compatibility, but these are secondary to the dominant long-term pattern of continuous system improvement and integration.

VII. Future Trends and Implications

6.1 Integrating Blockchain Technology

The development of smart contract management is related to the advancement of blockchain technology. As blockchain networks become more scalable, eco-friendly, and interconnected, smart contracts will become more advanced and widely used. Future developments might incorporate Layer 2 enhancements like sidechains and rollups to lower transaction costs and increase speed. Furthermore, smart contract systems for corporate use may be built on hybrid blockchains that are both private and public. With this integration, businesses will be able to preserve the advantages of decentralization—such as trust and unchangeability—while safeguarding confidential information within protected frameworks. The latter will allow stronger integration with systems of decentralized identity (DID) in which contracts can automatically determine and evaluate the reputation of the participants. The changes will not only make the execution more efficient, but also give more trust to the systems to share digital resources.

6.2 Exploitation of Smart Contract Technology in Different Business Sectors

The scope of smart contracts within the realms of finance, supply chain, and asset exchange is about to expand. In the coming time, healthcare, insurance, education, construction, and real estate will depend heavily on smart educational contracts to secure and automate their services. In healthcare, smart contracts would automate managing and tracking patient consent, billing, and the sharing of medical records. Smart contracts in insurance would automate the processing of claims to be executed instantly as soon as all conditions are met. They would solve latitudes and conflicts. Through smart contracts, educational institutions would have the ability to automate the verification of issued credentials in areas of transcript exchange and loan contracts. Even in agriculture, smart contracts would be applicable and used to conduct transparent farm-to-market transactions, subsidies, and ensure quality. Such systems are most likely to be extremely modular to enable users to write smartly coded contracts to achieve certain results without necessarily starting from scratch. This would increase access, besides making the usage more accessible to businesses that may be classified as small and medium in different industrial sectors. Comparing the two excerpts can help understand how integrating smart contracts into various areas enhances automation of internal procedures.

6.3 Regulatory Issues and Legal Consequences

Like any technology, smart contracts have peculiarities that can create legal and regulatory dilemmas. A question of whether the self-executing computer code is legally binding or not is one of the most essential issues. As these agreements are written in languages that are not legal, it may be exceedingly difficult for the legal system to discern the logic behind what is attempted, fairness, and breaches of contracts. The new legislation will need a transformation to acknowledge smart contracts as a legal tool that needs to be amended in legislation on contracts and different legal regulations on electronic agreements. The other option is that smart contract-enabled cross-border transactions could raise conflicting jurisdiction laws that require institutionalizing global standards or arbitration institutions. Issues related to Information Privacy, Data Protection, Consumers Rights, Taxation and even Liability in case of defects or unintended effects of computer programs should be expanded on further. Government agencies will be required to put in place compliance frameworks or marking certifying the transparency, accountability and consumer protection. To sum up, the aspect of regulation policies versus innovation will be very significant in the future. Concentrated on the correct areas, intelligent contracts will allow for easy implementation in the economy that becomes more and more automated, decentralized, and faster, where the interests of all involved parties are safeguarded.

VIII. Conclusion

To summarize, smart contract management is positioned as a novel form of enhancing the effectiveness, accountability, and trustworthiness of sharing agreements. This paper underscores the importance of smart contracts in automating processes such as executing agreements with transparency using authoritative ledgers and incorporating verification mechanisms that encourage trust among stakeholders. Illustrative case studies like library consortia, carsharing, and tool-sharing services showcase the tangible advantages and broad applicability of these technologies. Best practices for smart contracts such as establishing unambiguous term regulations, conducting audits, and open designs targetable for other systems for sharable modularity also strengthen the reliability of smart contract deployment. Smart contracts will be increasingly useful as more people integrate blockchain technology across multiple sectors like healthcare, education, and agriculture. They also need to develop laws that deal with how these concepts will be enforced, geographically governed, or liable. The future of resource sharing is in automated, decentralized, and legally binding frameworks which positions smart contract management as more than a technological advancement, but an essential decision in policy strategy. With these tools, organizations will foster

resilient, adaptable, and equitable sharing economies that respond to the realities of a digitally interlinked world.

REFERENCES

[1] Abd, W. H., & Kazem, D. A. (2022). The Relationship between the Balanced Scorecard and (ISO) Standards in Improving Banking Performance. *International Academic Journal of Business Management*, *9*(2), 40–47.

[2] Al-Bassam, M., Sonnino, A., Bano, S., Hrycyszyn, D., & Danezis, G. (2017). Chainspace: A sharded smart contracts platform. *arXiv preprint arXiv:1708.03778*.

[3] Atzei, N., Bartoletti, M., & Cimoli, T. (2017, March). A survey of attacks on ethereum smart contracts (sok). In *International conference on principles of security and trust* (pp. 164-186). Berlin, Heidelberg: Springer Berlin Heidelberg.

[4] Baggyalakshmi, N., Jokani, N. R., & Revathi, R. (2024). Managing and Billing Software for Hypermarket. *International Academic Journal of Innovative Research*, *11*(1), 17-26. https://doi.org/10.9756/IAJIR/V11I1/IAJIR1103

[5] Barile, J., Carreño, E., & Los Ríos-Escalante, D. (2024). *A review of mollusks farming in Chile*. *International Journal of Aquatic Research and Environmental Studies, 4*(1), 63–69. https://doi.org/10.70102/IJARES/V4I1/6

[6] Buterin, V. (2014). A next-generation smart contract and decentralized application platform. *white paper*, *3*(37), 2-1.

[7] Christidis, K., & Devetsikiotis, M. (2016). Blockchains and smart contracts for the internet of things. *IEEE access*, *4*, 2292-2303. https://doi.org/10.1109/ACCESS.2016.2566339

[8] Dabor, A., Aggreh, M., & Aneru, M. (2019). *Adoption of computerized accounting system by SMEs in Benin City*. *International Academic Journal of Economics, 6*(1), 123–140. https://doi.org/10.9756/IAJE/V6I1/1910009

[9] Delmolino, K., Arnett, M., Kosba, A., Miller, A., & Shi, E. (2016, February). Step by step towards creating a safe smart contract: Lessons and insights from a cryptocurrency lab. In *International conference on financial cryptography and data security* (pp. 79-94). Berlin, Heidelberg: Springer Berlin Heidelberg.

[10] Donkor, K., & Zhao, Z. (2023). Building Brand Equity Through Corporate Social Responsibility Initiatives. *Global Perspectives in Management*, *1*(1), 32-48.

[11] Frantz, C. K., & Nowostawski, M. (2016, September). 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)* (pp. 210-215). IEEE. https://doi.org/10.1109/FAS-W.2016.53

[12] Fu, Y., Li, C., Yu, F. R., Luan, T. H., Zhao, P., & Liu, S. (2022). A survey of blockchain and intelligent networking for the metaverse. *IEEE Internet of Things Journal*, *10*(4), 3587-3610.

[13] Gervais, A., Karame, G. O., Wüst, K., Glykantzis, V., Ritzdorf, H., & Capkun, S. (2016, October). On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security* (pp. 3-16). https://doi.org/10.1145/2976749.2978341

[14] Glaser, F. (2017). Pervasive decentralisation of digital infrastructures: *a framework for blockchain enabled system and use case analysis.*

[15] Heng, S., Aimtongkham, P., & So-In, C. (2023). A novel video-on-demand caching scheme using hybrid fuzzy logic least frequency and recently used with support vector machine. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications Dependable Applications*, *14*(1), 15-36. https://doi.org/10.58346/JOWUA.2023.I1.002

[16] Kondam, R. R., Sekhar, P. C., & Boya, P. K. (2024). Low Power SoC based Road Surface Crack Segmentation using Unet with Efficientnet-B0 Architecture. *Journal of VLSI Circuits and Systems*, *6*(1), 61-69. https://doi.org/10.31838/jvcs/06.01.11

[17] Kosba, A., Miller, A., Shi, E., Wen, Z., & Papamanthou, C. (2016, May). Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE symposium on security and privacy (SP)* (pp. 839-858). IEEE. https://doi.org/10.1109/SP.2016.55

[18] Kshetri, N. (2021). *Blockchain and sustainable supply chain management in developing countries*. *International Journal of Information Management, 60*, Article 102376. https://doi.org/10.1016/j.ijinfomgt.2021.102376

[19] Leiwi, A. A. A. (2022). *The repercussions of oil price fluctuations on Iraq's gross domestic product: Analytical study for the period between 2000–2019. International Academic Journal of Accounting and Financial Management, 9*(2), 48–61. https://doi.org/10.9756/ IAJAFM/V9I2/IAJAFM0906

[20] Luu, L., Chu, D. H., Olickel, H., Saxena, P., & Hobor, A. (2016, October). Making smart contracts smarter. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security* (pp. 254-269). https://doi.org/10.1145/2976749.2978309

[21] Mengelkamp, E., Gärttner, J., Rock, K., Kessler, S., Orsini, L., & Weinhardt, C. (2018). Designing microgrid energy markets: A case study: The Brooklyn Microgrid. *Applied energy*, *210*, 870-880. https://doi.org/10.1016/j.apenergy.2017.06.054

[22] Muralidharan, J. (2020). Wideband Patch Antenna for Military Applications. *National Journal of Antennas and Propagation*, *2*(1), 25-30. https://doi.org/10.31838/NJAP/02.01.05

[23] Pincheira, M., Donini, E., Vecchio, M., & Kanhere, S. (2022). A decentralized architecture for trusted dataset sharing using smart contracts and distributed storage. *Sensors*, *22*(23), 9118.

[24] Raghav, K., & Sunita, R. (2024). Advanced Material Selection and Structural Design for Sustainable Manufacturing of Automotive Components. *Association Journal of Interdisciplinary Technics in Engineering Mechanics*, *2*(3), 30-34.

[25] Rajput, D. S. (2024). Investigating the Role of Genetic Variants in Drug-Induced Liver Injury. *Clinical Journal for Medicine, Health and Pharmacy*, *2*(1), 30-39.

[26] Reyna, A., Martín, C., Chen, J., Soler, E., & Díaz, M. (2018). On blockchain and its integration with IoT. Challenges and opportunities. *Future generation computer systems*, *88*, 173-190. https://doi.org/10.1016/j.future.2018.05.046

[27] Savelyev, A. (2017). Contract law 2.0:'Smart'contracts as the beginning of the end of classic contract law. *Information & communications technology law*, *26*(2), 116-134. https://doi.org/10.1080/13600834.2017.1301036

[28] Sharukes, K., Vasudevan, R., Sivabalan, G., & Nagarajan, G. (2023). Defect Tracking System. *International Journal of Advances in Engineering and Emerging Technology*, *14*(1), 168-172.

[29] Sillaber, C., & Waltl, B. (2017). Life cycle of smart contracts in blockchain ecosystems. *Datenschutz und Datensicherheit-DuD*, *41*(8), 497-500.

[30] Sivaranjith, C., & Subramani, M. (2013). Development of reversible programmable gate array. *International Journal of Communication and Computer Technologies (IJCCTS)*, *1*(2), 72-78.

[31] Sredić, S., Knežević, N., & Milunović, I. (2024). *Effects of landfill leachates on ground and surface waters: A case study of a wild landfill in Eastern Bosnia and Herzegovina*. *Archives for Technical Sciences, 1*(30), 97–106. https://doi.org/10.59456/afts.2024.1630.097S