The Eastasouth Journal of Information System and Computer Science
Vol. 1, No. 01, August, pp. 219 - 230

Semantic Contract Meshes for Dynamic Partner Orchestration Across Multi-Enterprise Cloud Ecosystems

Shivakrishna Vangala
Principal Architect, T-Mobile USA Inc, Bellevue, USA

Article Info

Article history:

Received Aug, 2023
Revised Aug, 2023
Accepted Aug, 2023

Keywords:

Knowledge Graph;
Multi-Enterprise Cloud;
Ontology;
Partner Orchestration;
Petri Nets;
Semantic Contract Mesh;
Service Level Agreement;
Smart Contracts;
Zero Trust

ABSTRACT

Multi-enterprise cloud ecosystems need a coordinated and accelerated approach to onboard, coordinate, and replace partners, that conventional bilateral integration agreements cannot keep up with. This systematic literature review aims to describe and assess the semantic contract mesh, a decentralized layer that is characterized by the binding of machine-interpretable agreements to shared ontologies, enforcement through zero-trust policy points, and execution through verifiable workflow models, and that has been published from 1990 to 2021. Sources were rated on six weighted criteria and put into 7 themes with five having at least 4 sources. Three latency regimes are suggested along with a formal contract tuple, a weighted Jaccard compatibility measure and a Petri net state equation. Comparative scoring gives the mesh a weighted 2.75 out of 3.00 points while point-to-point integration gets a 0.65. With an illustrative parameterization, mesh latency is 38.0ms at 50 partners, while pairwise integration takes 1,225ms. Latencies and scores are analytic, not experimental. In just 8 of the 42 theme–capability cells, the maximum code is achieved, revealing fragmentation across the literature. Ethical, regulatory and scalability challenges are explored, and a phased, ontology-first approach to adoption for enterprises seeking to incrementally move from bilateral integration to shared machine-interpretable agreements is proposed.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Name: Shivakrishna Vangala
Institution: Principal Architect, T-Mobile USA Inc, Bellevue, USA
Email: shivakrishna.vangala@gmail.com

1. INTRODUCTION

Cloud computing has moved from the single-provider hosting to a substrate shared among independent enterprises that jointly provide products, share data and create services. The underlying characteristics of the field focused on elasticity, pay-per-use pricing, and the fact that the services could seem limitless, and identified ten barriers to adoption, including service availability, data lock-in, and unpredictable performance [1]. Complex multi-organizational scenarios where each organization has its own governance are amplified by each obstacle – no one entity has control over the infrastructure that the business result relies upon.

The agreements with such partners are traditionally negotiated as a prose document, and then manually translated into integration code. While automation

eliminates that manual process, it also brings in new risks: twelve types of vulnerabilities in smart contracts have been identified by a survey of attacks on the Ethereum network, ranging from vulnerabilities in the language and virtual machine to vulnerabilities in the blockchain [2]. If there is no common and visible understanding of the meaning and interpretation of any given term in a proposed agreement, then the agreement is made quickly, but not guaranteed, and this drives the need for a machine-readable and semantically-aware representation.

The vision of the Semantic Web, which was proposed, was that data marked with explicit meaning would enable software agents to perform complex operations in a cross-organizational manner [3]. The same idea, when applied to commercial agreements, implies that obligations, permissions, prices, and service levels can be described against common ontologies, which means that partners can have the same interpretation of the terms of an agreement in spite of their different internal languages. The problem this review is dealing with is that there is no integrated architecture that brings together semantic description, formal contracts and runtime orchestration for dynamic partner management.

And there is a precedent in container orchestration, as cluster managers have moved from a centralised scheduler to a declarative, controller based systems which constantly check the desired state against the observed state. An analogous reconciliation concept is suggested for partnerships: a network of semantic contracts defines what's expected of the partnership, while automatic controllers monitor if things are getting out of sync, renegotiate terms, and replace other partners. The work of this review includes a coded synthesis of 29 sources, a layered mesh architecture, a series of formal models and a comparative evaluation including identified barriers.

2. REVIEW METHODOLOGY

The review was done to a protocol adapted form systematic review practice. Only peer-reviewed articles, book chapters, standards documents and one textbook on cloud delivery, semantic technologies, service computing, contract formalization, workflow theory, and blockchain were included in the corpus. First conceptual boundary for the Cloud dimension was evolving from the treatment of computing as a fifth utility [4]. Overall, 29 sources were retained over a 31-year time span, and Figure 1 sees their distribution over five publication years, including 18 of 29 sources for the 2000-2019 period. Standards documents were only retained if they provided normative definitions which the other sources made assumptions about, and sources were only screened for relevance to cross-organizational coordination and were not for single-enterprise optimization.

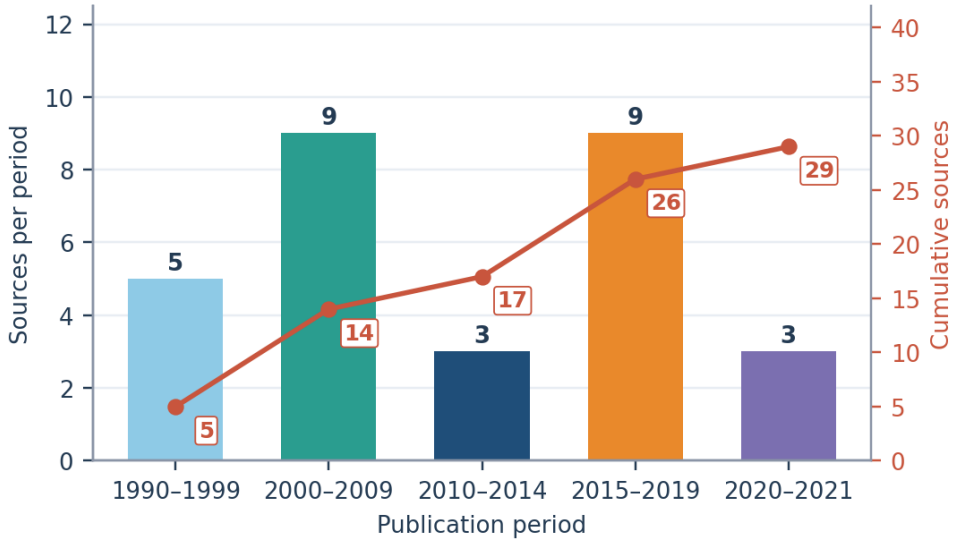


Figure 1. Distribution of Reviewed Sources by Publication Period

Coding against these 6 criteria was done on an ordinal 4-point scale ranging from not addressed (0) to central contribution (3). The criteria encompass decentralised implementation of agreements, which is exemplified in Internet of Things contexts with smart contracts on the blockchain [5]. The average score for the specific approach j is derived in the following manner: candidate approach j is scored by adding weights wk for each criterion k and assigning a code cjk to each, with the weights being the sum of one. The six criteria and their weights are presented in Table 1.

$$S_j = \sum_{k=1}^{K} w_k c_{jk} \qquad (1)$$

Table 1. Criteria and Weights Used for Evidence Coding

Code	Criterion	Weight
C1	Semantic interoperability	0.20
C2	Contract formalization	0.20
C3	Policy enforcement	0.15
C4	Dynamic reconfiguration	0.20
C5	Auditability	0.10
C6	Scalability	0.15
Total		**1.00**

Note: Codes range from 0 (not addressed) to 3 (central contribution).

3. LITERATURE SYNTHESIS

3.1 Service and Process Foundations

The history of the microservice style goes back to previous service-oriented practice, emphasizing decentralized data handling and independent scaling [6]. With a contract mesh, obligation is naturally tied to the microservice since each service can be versioned, replaced, and monitored, and without affecting the rest of the partner estate.

Business process management provides the language of the lifecycle for modelling, executing, monitoring and improving cross-organizational work, and process models serve as the "executable" link between agreements and operations [7].

A study on web service composition was conducted by differentiating static and dynamic composition (bound at design time and discovered at run-time, respectively), and showed that semantic description and quality of service are prerequisites

for dynamic composition [8]. This requirement is carried forward to dynamic partner orchestration. If there are no explicit semantics and measurable quality attributes, a replacement partner cannot be automatically chosen and the promise of cloud elasticity for reconfiguration speed is broken at the organizational boundary. Table 2 shows the seven themes found in this review and their respective sources and roles within the mesh.

Table 2. Mapping of Reviewed Literature to Concepts and Mesh Roles

Theme	Sources	Core concept	Role in the mesh
Cloud and infrastructure	[1], [4], [9], [10], [11]	Elastic, metered, interconnected cloud	Infrastructure layer and federation
Service computing	[6], [8], [12], [13], [14]	Services, composition, interfaces	Capability units and bindings
Semantic and knowledge	[3], [15], [16], [17], [18]	Ontologies, graphs, mapping	Shared meaning and alignment
Contracts and policy	[19], [20], [21], [22]	Deontic clauses, SLAs, zero trust	Clause model and enforcement
Process and workflow	[7], [23], [24]	Process models, Petri nets	Orchestration of lifecycle
Blockchain	[2], [5], [25]	Smart contracts, ledger limits	Anchoring and audit trail
Ecosystems, agents, data spaces	[26], [27], [28], [29]	Platforms, complementarity, agents	Governance context and actors

Source: Synthesized from the reviewed references.

3.2 Semantics, Ecosystems, and Agents

An ontology was defined as an explicit specification of a conceptualization, a definition that established shared vocabulary as an engineering artifact [15]. In the mesh, ontologies fix the meaning of terms such as delivery, availability, and personal data. Compatibility between contracts a and b is expressed by Equation (2), a weighted Jaccard measure in which $O_a(i)$ is the set of ontology concepts used by contract a in facet i, F is the number of facets, and λ_i is a facet weight.

$$\mathrm{sim}(a,b) = \sum_{i=1}^{F} \lambda_i \frac{|O_a(i) \cap O_b(i)|}{|O_a(i) \cup O_b(i)|} \quad (2)$$

Digital platform ecosystems were defined as a sociotechnical arrangement in which a platform owner and complementors create and share value together by leveraging boundary resources and by developing a shared governance framework [26]. A contract mesh is one layer below this governance, as it is an agreement of the participants of the ecosystem that is binding and verifiable, and doesn't contain any bias towards the owner of the platform.

Knowledge graphs are graph-structured data with schema, identity and context, which are accompanied by graph validation, embedding and reasoning techniques, to constitute the storage and inference layer for ontologies and contract instances [16]. Graph queries are used to answer questions, like which partners have a currently certified capability within a particular jurisdiction; shape constraints serve to validate instances of a contract against the governing shape.

According to the theory of ecosystems developed by Jacobides and his colleagues [27] modularity and non-generic complementarities are the characteristics of ecosystems, with partners being interdependent, and not owned. This view is valid in the sense that complementor relationships are diverse and dynamic. As seen in Figure 2, 5 of the 7 themes have more than 4 sources while 3 for each theme of process and blockchain.

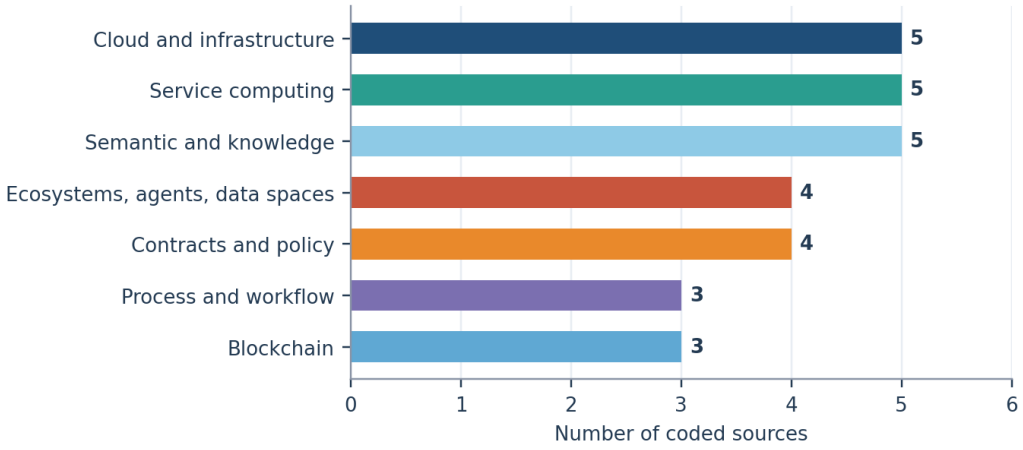


Figure 2. Distribution of Reviewed Sources Across Seven Themes

Through agent research, autonomy, social ability, reactivity and pro-activeness were identified as the properties of software that work on behalf of an organization, and negotiation and coordination were recognized as key issues [28]. The proposal for autonomous partner agents is that they will be the mesh actors which publish capability offers, evaluate counterproposals based on local policy, and trigger renegotiation when monitored conditions differ from the agreement state. The numerical anchors extracted have been combined into Table 3.

Table 3. Numerical Anchors Extracted from the Reviewed Corpus

Source	Quantitative anchor	Use in the mesh
[1]	10 obstacles to cloud adoption	Assurance criteria
[2]	12 vulnerability types in 3 levels	Verification scope
[4]	Computing as the 5th utility	Metered contract pricing
[10]	5 characteristics, 3 service models, 4 deployment models	Infrastructure taxonomy
[22]	7 zero-trust tenets	Enforcement checks
Corpus	29 sources, 31-year span, 7 themes	Coding frame

Source: Compiled from the reviewed references.

3.3 Policy, Service Level, and Semantic Services

A policy language for pervasive environment, based on the semantic web technology, that formalized rights, prohibitions, obligations and dispensations, and supported delegation and conflict resolution between policies [19]. Such deontic constructions provide the lexical elements for contract clauses in the mesh. The measurable predicates used in the following service level specification are provided by the service.

The WSLA framework defined machine-processable service level agreements between parties, service objects, and obligations, as well as metrics, measurement instructions, and actions if the SLA is violated [20]. The compliance is summarized over an evaluation period in Equation (3) where V denotes the number of violated service level objectives while N denotes the number of the evaluated service level objectives. These ratios are taken into consideration when making renegotiation decisions and if a ratio is consistently below an agreed floor over multiple periods then it will result in the responsible partner being replaced.

$$\Gamma = 1 - \frac{V}{N} \qquad (3)$$

In order to automate service discovery, invocation, composition and monitoring, the Semantic Web services were proposed to be described in computer-interpretable markup language, allowing the agents to reason about the service functionality [12]. The mesh follows it for partner capabilities, enabling a new partner to be matched up with the requirements in the contract without prior bilateral engineering. In terms of theme and capability, as illustrated in Figure 3, only 8 of 42 theme – capability cells achieve the maximum code of 3, and no theme does more than two.

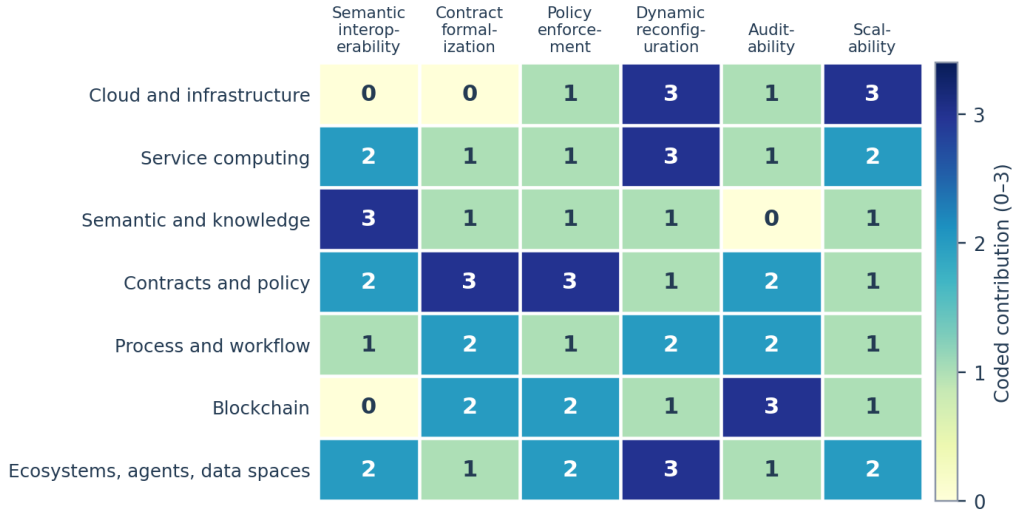


Figure 3. Coded Contribution of Each Theme to Mesh Capabilities

4. SEMANTIC CONTRACT MESH FRAMEWORK

4.1 Layered Architecture

Architecture consists of five layers, as shown in Figure 4. The lowest layer takes over the 3 service models and 4 deployment models of the NIST definition as well as its five essential characteristics of on-demand self-service, broad network access, resource pooling, rapid elasticity, and measured service [10]. Higher layers provide semantic description, contract management, orchestration and assurance, and zero-trust enforcement is across all layers. This separation enables each layer to develop separately, without having to redeploy the infrastructure layer, for instance, if a new version of the ontology is developed.

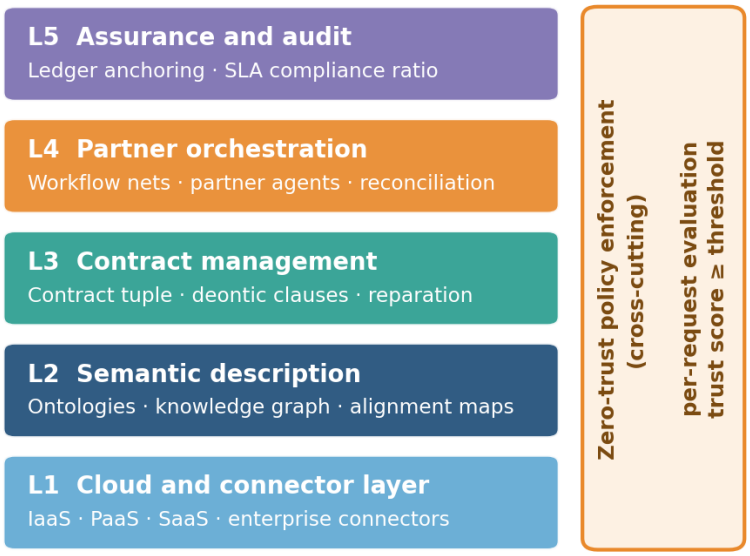


Figure 4. Layered Architecture of the Semantic Contract Mesh

The International Data Spaces case proved that a multi-sided platform allows the integration of trusted connectors, shared vocabularies and usage-control policies to facilitate sovereign data exchange without any central data owner [29]. The mesh has the same connectivity pattern: Each enterprise runs its own endpoint with capability descriptions, which determines whether a contract proposal is accepted or rejected, and imposes local usage terms before anything crosses its boundary. Partner onboarding becomes as simple as publishing a capability description and signing a contract proposal since the connector is the only place of interaction.

Service-oriented computing was explained in terms of the layered concerns of description, discovery, composition and management, and open challenges of dynamic reconfiguration, quality-aware composition and cross-organizational governance [13]. The concerns are aligned to the mesh layers, and they allow for reusing existing abstractions of service-oriented interactions instead of devising new ones and lowering the learning curve for partner engineering teams.

According to [14] a comparison between RESTful and WSDL-based services resulted in the conclusion that the decision on which architecture is used is based on requirements like interface flexibility, transactional semantics, and tooling rather than a technical superiority. Accordingly, the mesh is interface-agnostic, that is, contract clauses are bound to semantic capability descriptions and the transport style – resource-oriented or message-oriented – is chosen per partner.

4.2 Formal Models

A formal language for electronic contracts was introduced to define obligations and permissions and forbidions as well as reparation clauses, with a trace semantics for the analysis of conflicts [21]. A mesh contract is described as the tuple in Equation (4), which consists of the parties P, clauses K, ontology bindings O, reparation rules R, and monitoring predicates M; the relationship between the type of clauses and modalities and enforcement points are shown in Table 4.

$$C = \langle P, K, O, R, M \rangle \quad (4)$$

Table 4. Contract Clause Types, Modalities, and Enforcement Points

Clause type	Modality	Example	Enforcement point	Basis
Obligation	Must	Deliver capacity within agreed latency	SLA monitor and ledger	[20], [21]
Permission	May	Read a declared data class	Policy enforcement point	[19]
Prohibition	Must not	Export data outside jurisdiction	Policy enforcement point	[19], [22]
Reparation	Must, if violated	Issue service credit after breach	Contract controller	[20], [21]
Delegation	May transfer	Substitute partner assumes duty	Orchestration layer	[19]

Source: Synthesized from the reviewed references.

Some of the key principles of Zero trust architecture include per-session access, dynamic policy, and continuous monitoring of asset posture [22]. Enforcement in the mesh is similar: Access is allowed for each partner request at a policy enforcement point if the weighted posture score meets Equation (5), where si is a posture signal, wi is the weight of the signal, m is the number of signals, and θC is the threshold defined by the contract C.

$$\sum_{i=1}^{m} w_i s_i \geq \theta_C \quad (5)$$

The federated database research focused on schematic and semantic heterogeneity of autonomous, distributed systems using the layered schemas to maintain the autonomy of the local systems while providing integrated views [17]. The same principle is used in the mesh for ontologies: there is a local vocabulary for each partner, and translations of local terms to a common reference vocabulary (alignment mappings) yield a measure in Equation (2) that is computed on translated, not raw concept sets.

The knowledge engineering paradigm shifted from knowledge transfer to model construction, and ontologies and problem-solving methods were regarded as reusable components [18]. Building a mesh ontology is therefore an iterative process that involves scoping, reusing existing vocabularies, formalizing, and testing the generated artifacts using competency questions while keeping contracts to avoid any semantic drift between partners.

5. RESULTS AND DISCUSSION

5.1 Comparative Assessment of Integration Approaches

A taxonomy of interrelated cloud environments was described where providers work together based on mutual agreements, and identified key challenges such as interoperability, service level negotiation, and trust in multi-cloud environments [11]. The results of the five approaches for equating using Equation (1) are compared on the six criteria in Table 5. The mesh's weighted score is 2.75, while federated schemas score 1.35 and point-to-point integration scores 0.65. Federated schemas achieve the best results for interoperability and scalability of alternatives, while blockchain-only contracts are best in terms of auditability and are lack of semantic grounding. The scores are analytic codes given by the review protocol, not experimental measures.

Table 5. Comparative Scores of Integration Approaches

Approach	SI	CF	PE	DR	AU	SC	Score
Point-to-point integration	0	1	1	1	1	0	0.65
Centralized broker or ESB	1	1	2	1	2	1	1.25
Federated schema integration	2	1	1	1	1	2	1.35
Blockchain-only contracts	0	2	2	1	3	1	1.35
Semantic contract mesh	3	3	3	3	2	2	2.75

Note: SI = semantic interoperability; CF = contract formalization; PE = policy enforcement; DR = dynamic reconfiguration; AU = auditability; SC = scalability. Score is computed with Equation (1).

5.2 *Orchestration and Latency Evaluation*

Petri nets offer a formal basis for workflow specification, including a graphical notation, state-based semantics and analysis of properties of workflows, including soundness and deadlock [23]. To model each partnership lifecycle, the orchestration layer is then represented as a workflow net that will only fire if the input places contain tokens. The marking is done according to Equation (6), where M is the marking vector, N the incidence matrix and σ the firing count vector. Figure 5 shows an onboarding network having 4 places and 3 transitions. The soundness of such a net ensures that eventually all onboarding cases are transitioned to the active state, while having no remaining tokens, a property that should be satisfied before a contract is admitted to the mesh.

$$M' = M + N\sigma \qquad (6)$$

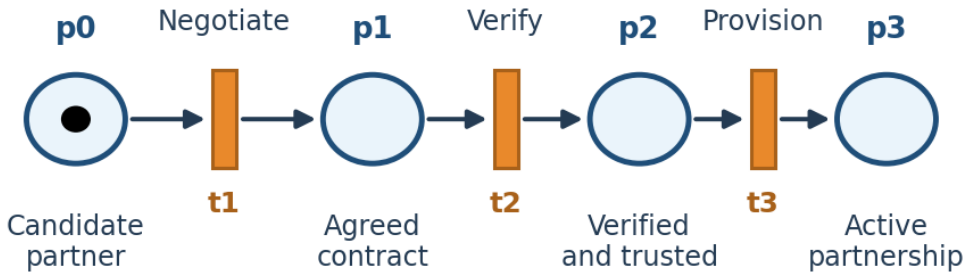


Figure 5. Workflow Net for Partner Onboarding

Collaborative processes among mutually distrustful participants were shown to be monitored by compiling process models into smart contracts to check whether the messages were sent and the processes had been executed, and the states had been changed appropriately on a ledger [24]. The ledger anchors contract hashes and state transitions and not payloads in the mesh.

The mesh achieves a lower latency than pairwise starting from n = 8 and is lower than the centralized broker from n = 15, as shown in the Table 6 and Figure 6 with the parameters α = 12 ms, β = 1.2 ms, γ = 1.0 ms, δ = 4.5 ms, and ε = 11 ms.

$$L_c(n) = \alpha + \beta n \quad (7)$$

$$L_p(n) = \gamma \frac{n(n-1)}{2} \quad (8)$$

$$L_m(n) = \delta \lceil \log_2 n \rceil + \varepsilon \quad (9)$$

Table 6. Illustrative Coordination Latency Under Three Regimes

Partners (n)	Broker L_c (ms)	Pairwise L_p (ms)	Mesh L_m (ms)	Pairwise/Mesh ratio
5	18.0	10.0	24.5	0.4
10	24.0	45.0	29.0	1.6
20	36.0	190.0	33.5	5.7
50	72.0	1,225.0	38.0	32.2
100	132.0	4,950.0	42.5	116.5

Note: Values are computed from Equations (7) to (9) with illustrative parameters and are not empirical measurements.

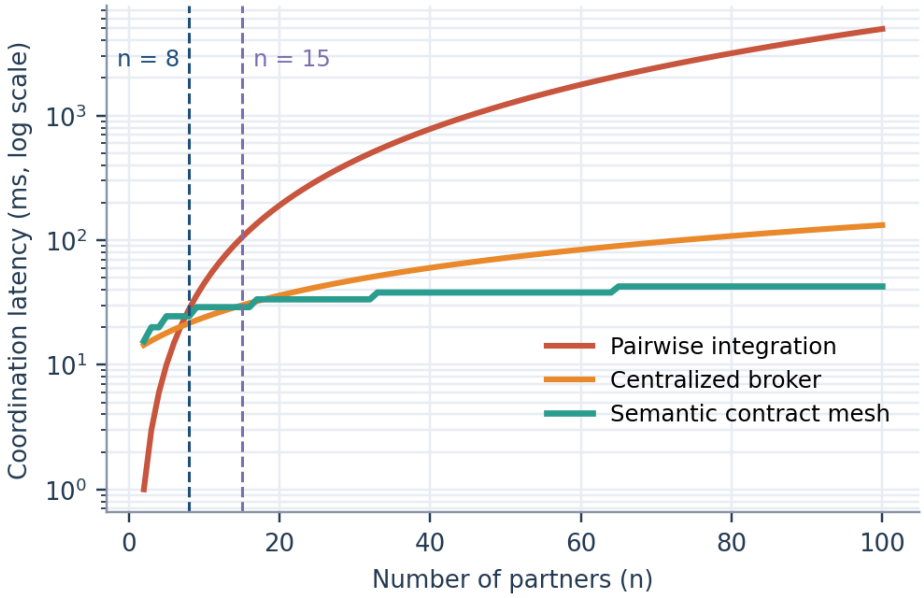


Figure 6. Coordination Latency Versus Number of Partners

5.3 Barriers, Ethics, and Regulation

In a survey of challenges in blockchain, scalability, privacy leakage, selfish mining were pointed out as the major challenges, while opportunities in decentralized trust were mentioned [25]. The results of these findings limit the role of the ledger to an anchoring role as assumed above. Other obstacles outside the ledger exist. The governance of shared vocabularies is required for semantic alignment, personal data appears in contract terms, which imposes a data-protection obligation, and machine-executed clauses have different legal status in different jurisdictions. Ethical issues are the unclear renegotiation by automated systems and the imbalance of bargaining power between the platform owners and the complementors. This is why it is recommended to take the following stages: standardization of vocabulary first, contracts under supervision second, and contracts independently negotiated third.

6. CONCLUSION

This systematical review of 29 sources confirmed that semantic contract meshes have been defined as a layered architecture; ontologies are used to specify the meaning of clauses, formal contract tuples to represent obligations, permissions and prohibitions are specified for each request, and workflow nets to orchestrate the partnership lifecycle is controlled by zero trust enforcement. Comparative coding revealed that the mesh was the most highly weighted of five different ways to integrate, and the illustrative latency model suggested that the mesh scales logarithmically while pairwise scales quadratically. Literature is still fragmented as no theme treats more than two capabilities at the maximum level. Future research should expand on the coded scores by having them reviewed by experts panels, test latency on prototype examples, create common ontologies for the regulatory obligations, and explore regulatory governance to reduce the ability for opaque automated renegotiation.

ACKNOWLEDGEMENTS

The authors thank the Eastasouth Institute for institutional support during the preparation of this revie

REFERENCES

[1] M. Armbrust *et al.*, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010, doi: 10.1145/1721654.1721672.

[2] 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: https://doi.org/10.1007/978-3-662-54455-6_8.

[3] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Sci. Am.*, vol. 284, no. 5, pp. 34–43, 2001, doi: 10.1038/scientificamerican0501-34.

[4] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Futur. Gener. Comput. Syst.*, vol. 25, no. 6, pp. 599–616, 2009, doi: https://doi.org/10.1016/j.future.2008.12.001.

[5] K. Christidis and M. Devetsikiotis, "Blockchains and Smart Contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016, doi: 10.1109/ACCESS.2016.2566339.

[6] N. Dragoni *et al.*, "Microservices: Yesterday, Today, and Tomorrow," in *Present and Ulterior Software Engineering*, M. Mazzara and B. Meyer, Eds., Cham: Springer International Publishing, 2017, pp. 195–216. doi: 10.1007/978-3-319-67425-4_12.

[7] M. Dumas, M. La Rosa, J. Mendling, and H. A. Reijers, *Fundamentals of Business Process Management*, 2nd ed. Springer, 2018. doi: 10.1007/978-3-662-56509-4.

[8] S. Dustdar and W. Schreiner, "A survey on web services composition," *Int. J. Web Grid Serv.*, vol. 1, no. 1, pp. 1–30, 2005, doi: 10.1504/IJWGS.2005.007545.

[9] B. Burns, B. Grant, D. Oppenheimer, E. Brewer, and J. Wilkes, "Borg, Omega, and Kubernetes: Lessons learned from three container-management systems over a decade," *Queue*, vol. 14, no. 1, pp. 70–93, Jan. 2016, doi: 10.1145/2898442.2898444.

[10] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," *Natl. Inst. Stand. Technol. Spec. Publ. 800-145*, 2011, doi: https://doi.org/10.6028/NIST.SP.800-145.

[11] A. N. Toosi, R. N. Calheiros, and R. Buyya, "Interconnected Cloud Computing Environments: Challenges, Taxonomy, and Survey," *ACM Comput. Surv.*, vol. 47, no. 1, pp. 1–47, 2014, doi: 10.1145/2593512.

[12] S. A. McIlraith, T. C. Son, and H. Zeng, "Semantic Web services," *IEEE Intell. Syst.*, vol. 16, no. 2, pp. 46–53, 2001, doi: 10.1109/5254.920599.

[13] M. P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann, "Service-Oriented Computing: State of the Art and Research Challenges," *Computer (Long. Beach. Calif).*, vol. 40, no. 11, pp. 38–45, 2007, doi: 10.1109/MC.2007.400.

[14] C. Pautasso, O. Zimmermann, and F. Leymann, "Restful web services vs. 'big'' web services: making the right architectural decision," in *Proceedings of the 17th International Conference on World Wide Web*, in WWW '08. New York, NY, USA: Association for Computing Machinery, 2008, pp. 805–814. doi: 10.1145/1367497.1367606.

[15] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowl. Acquis.*, vol. 5, no. 2, pp. 199–220, 1993, doi: https://doi.org/10.1006/knac.1993.1008.

[16] A. Hogan *et al.*, "Knowledge graphs," *ACM Comput. Surv.*, vol. 54, no. 4, pp. 1–37, 2021, doi: 10.1145/3447772.

[17] A. P. Sheth and J. A. Larson, "Federated database systems for managing distributed, heterogeneous, and autonomous databases," *ACM Comput. Surv.*, vol. 22, no. 3, pp. 183–236, Sep. 1990, doi: 10.1145/96602.96604.

[18] R. Studer, V. R. Benjamins, and D. Fensel, "Knowledge engineering: Principles and methods," *Data Knowl. Eng.*, vol.

25, no. 1, pp. 161–197, 1998, doi: https://doi.org/10.1016/S0169-023X(97)00056-6.

[19] L. Kagal, T. Finin, and A. Joshi, "A policy language for a pervasive computing environment," in *Proceedings POLICY 2003. IEEE 4th International Workshop on Policies for Distributed Systems and Networks*, 2003, pp. 63–74. doi: 10.1109/POLICY.2003.1206958.

[20] A. Keller and H. Ludwig, "The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services," *J. Netw. Syst. Manag.*, vol. 11, no. 1, pp. 57–81, 2003, doi: 10.1023/A:1022445108617.

[21] C. Prisacariu and G. Schneider, "A Formal Language for Electronic Contracts BT - Formal Methods for Open Object-Based Distributed Systems," in *Formal Methods for Open Object-Based Distributed Systems*, M. M. Bonsangue and E. B. Johnsen, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 174–189. doi: https://doi.org/10.1007/978-3-540-72952-5_11.

[22] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero trust architecture," *Natl. Inst. Stand. Technol. Spec. Publ. 800-207*, 2020, doi: https://doi.org/10.6028/NIST.SP.800-207.

[23] W. M. P. Van der Aalst, "The application of Petri nets to workflow management," *J. Circuits Syst. Comput.*, vol. 8, no. 1, pp. 21–66, 1998, doi: https://doi.org/10.1142/S0218126698000043.

[24] I. Weber, X. Xu, R. Riveret, G. Governatori, A. Ponomarev, and J. Mendling, "Untrusted Business Process Monitoring and Execution Using Blockchain BT - Business Process Management," in *Business Process Management*, M. La Rosa, P. Loos, and O. Pastor, Eds., Cham: Springer International Publishing, 2016, pp. 329–347. doi: https://doi.org/10.1007/978-3-319-45348-4_19.

[25] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *Int. J. web grid Serv.*, vol. 14, no. 4, pp. 352–375, 2018, doi: https://doi.org/10.1504/IJWGS.2018.095647.

[26] A. Hein *et al.*, "Digital platform ecosystems," *Electron. Mark.*, vol. 30, no. 1, pp. 87–98, 2020, doi: 10.1007/s12525-019-00377-4.

[27] M. G. Jacobides, C. Cennamo, and A. Gawer, "Towards a theory of ecosystems," *Strateg. Manag. J.*, vol. 39, no. 8, pp. 2255–2276, Aug. 2018, doi: https://doi.org/10.1002/smj.2904.

[28] N. R. Jennings, K. Sycara, and M. Wooldridge, "A Roadmap of Agent Research and Development," *Auton. Agent. Multi. Agent. Syst.*, vol. 1, no. 1, pp. 7–38, 1998, doi: 10.1023/A:1010090405266.

[29] B. Otto and M. Jarke, "Designing a multi-sided data platform: findings from the International Data Spaces case," *Electron. Mark.*, vol. 29, no. 4, pp. 561–580, 2019, doi: 10.1007/s12525-019-00362-x.