International Journal of Scientific Research & Engineering Trends
Volume 5, Issue 6, Nov-Dec-2019, ISSN (Online): 2395-566X

Policy-Driven Automation for Scalable Governance in Enterprise Big Data Platforms

Madhava Rao Thota
Infra. Technology Specialist

Abstract - As enterprise data platforms continue to expand in scale, diversity, and operational criticality, the combination of high data velocity, exponential data growth, and increasingly stringent regulatory requirements renders manual governance and ad hoc operational controls both inefficient and error-prone. In response to these pressures, policy-driven automation has emerged as a foundational paradigm for managing end-to-end data lifecycles, fine-grained access control, regulatory compliance, and repeatable operational workflows across heterogeneous, distributed environments. This article synthesizes prior academic research and industry practices published between 2000 and 2018 to examine how declarative, machine-interpretable policies can be systematically translated into automated enforcement actions within modern enterprise data platforms. Drawing on established policy frameworks, rule-oriented and distributed data management systems, and workflow orchestration engines, we present an integrated architectural perspective that spans policy definition, policy decision evaluation, and policy execution. The discussion is grounded in practical, widely adopted database and Big Data technologies including MongoDB, Apache Cassandra, and DataStax Enterprise illustrating how policy-driven automation enables scalable governance, operational resilience, and auditable compliance while preserving the flexibility and performance required by contemporary enterprise data ecosystems.

Keywords - Policy-driven automation; Enterprise data platforms; Data governance; Big Data; MongoDB; Apache Cassandra; DataStax; XACML; iRODS; DataOps; Workflow orchestration.

INTRODUCTION

Enterprise data platforms have undergone a profound transformation over the past two decades, shifting from tightly controlled, centralized relational database systems to highly distributed, polyglot environments that integrate transactional databases, Big Data stores, streaming systems, and cloud-native analytics platforms. This evolution has been driven by the need to support massive data volumes, real-time processing requirements, and globally distributed users. Technologies such as MongoDB and Apache Cassandra exemplify this shift by offering flexible schemas, horizontal scalability, and built-in fault tolerance, enabling enterprises to handle diverse workloads ranging from operational transactions to large-scale analytics. However, these architectural advantages come at the cost of increased operational complexity, particularly in areas such as data consistency management, access control, replication configuration, and lifecycle governance. As data estates grow across clusters, regions, and cloud environments, maintaining a coherent governance posture becomes increasingly challenging. The proliferation of heterogeneous tools and platforms further complicates coordination, visibility, and control. Consequently, enterprises face mounting risks related to data sprawl, misconfiguration, and regulatory non-compliance if governance mechanisms fail to evolve alongside platform scale.

Traditional approaches to managing enterprise data platforms have relied heavily on procedural techniques, including manual scripts, static configuration files, and operational runbooks maintained by specialized teams. While these methods may suffice in small or stable environments, they do not scale effectively in dynamic, distributed systems where change is frequent and system state is constantly evolving. Manual processes are inherently error-prone and difficult to audit, particularly when governance requirements must be enforced consistently across multiple platforms and organizational boundaries. Ad hoc controls often emerge organically to address immediate operational needs, leading to fragmented solutions that are hard to reason about holistically. Siloed operational processes further exacerbate the problem by isolating governance responsibilities within individual teams or systems, reducing transparency and coordination. As a result, enterprises struggle to ensure that policies related to security, data retention, availability, and compliance are applied uniformly. These limitations highlight the need for a more systematic, scalable approach to governance and operational control.

© 2019 IJSRET

Policy-driven automation addresses these challenges by externalizing governance logic from application code and low-level operational tooling into explicit, declarative policies that express organizational intent. Rather than embedding enforcement logic directly into scripts or services, policies specify what must be enforced such as access constraints, retention periods, replication strategies, or regulatory obligations using formal, machine-interpretable rules. Automation frameworks and policy engines are then responsible for determining how these policies are evaluated and enforced at runtime. This separation of intent from mechanism enables governance requirements to evolve independently of underlying system implementations. By decoupling policy definition from execution, policy-driven systems improve adaptability to changing regulations and business needs while enhancing auditability and traceability. Consistent policy evaluation across platforms also reduces configuration drift and operational inconsistency. In large-scale data environments, this approach provides a foundation for resilient, transparent, and scalable governance that aligns technical operations with organizational and regulatory objectives.

II. CONCEPTUAL FOUNDATIONS OF POLICY-DRIVEN DATA MANAGEMENT

Early research on automated data management emphasized the need for self-managing systems capable of operating reliably under dynamic workloads, partial failures, and evolving application demands. As data volumes increased and distributed architectures became more prevalent, researchers recognized that static configuration and manual intervention were insufficient to maintain performance, availability, and correctness. This led to the exploration of autonomic computing principles, where systems could monitor their own state, reason about deviations from desired behavior, and take corrective actions automatically. Policy-based approaches emerged as a practical mechanism to encode high-level administrative intent in a form that machines could interpret and act upon. By introducing rule engines and decision layers, these systems could continuously evaluate system state against predefined constraints. This allowed automated responses to workload fluctuations, node failures, or changes in resource availability. Importantly, policies provided a level of abstraction above low-level configuration parameters, enabling administrators to reason in terms of objectives rather than implementation details. This shift laid the groundwork for scalable automation in increasingly complex data environments.

Within distributed data systems, policy-based mechanisms have been applied to a broad range of governance and operational concerns, including data placement, replication, preservation, and access control. In large-scale environments, decisions about where data should reside, how many replicas should exist, and when data should be migrated or archived are influenced by performance requirements, cost constraints, and regulatory obligations. Encoding these considerations as policies allows systems to make consistent, repeatable decisions across diverse infrastructure components. Policies have also been used to enforce access restrictions and data handling rules, ensuring that sensitive information is protected regardless of where it is stored or processed. In preservation and archival contexts, policy-driven systems can automatically manage retention periods, integrity checks, and format migrations over long time horizons. These capabilities are especially valuable in environments where data lifecycles span years or decades. By embedding such rules into the management fabric of distributed systems, policy-based approaches reduce operational burden while improving reliability and compliance.

A key architectural insight from this body of work is the clear separation of policy definition, policy decision, and policy enforcement. Policy definition focuses on expressing organizational or regulatory intent in a declarative form, independent of specific technologies or deployment details. Policy decision mechanisms are responsible for interpreting these definitions in light of current system state and contextual information, determining which actions should be taken. Policy enforcement components then carry out those actions within the underlying infrastructure, such as adjusting configurations, triggering workflows, or denying access requests. This separation enables organizations to evolve governance requirements driven by regulatory change, business priorities, or risk posture without redesigning core data infrastructure. The approach is particularly relevant for Big Data platforms, where operational logic must span multiple storage engines, clusters, and geographic regions. By decoupling intent from execution, policy-based architectures support adaptability, scalability, and long-term sustainability in complex enterprise data ecosystems.

Policy-Driven Data Management Architecture

Figure 1 illustrates a representative policy-driven data management architecture, exemplified by the Integrated Rule-Oriented Data System (iRODS), which operationalizes governance through explicit rules and automated enforcement mechanisms. In this model, policies are expressed as declarative rules that capture administrative intent, compliance requirements, and operational constraints at a high level of abstraction. A centralized rule engine continuously evaluates

© 2019 IJSRET

these policies against system events and contextual metadata, such as data creation, access attempts, or lifecycle transitions. When predefined conditions are satisfied, the engine orchestrates corresponding actions by invoking modular microservices. These microservices encapsulate concrete operational logic, including data movement, replication, validation, or notification tasks.

The separation between rule evaluation and execution allows policies to be modified without altering underlying services. As a result, governance behavior can evolve dynamically in response to regulatory changes or business needs. This architecture also supports fine-grained auditing by logging policy evaluations and enforcement outcomes. Collectively, these capabilities demonstrate how policy-driven systems can transform governance from a manual oversight function into a programmable, automated control plane.

This architectural pattern has direct applicability to modern enterprise Big Data platforms, where scale and distribution significantly increase operational complexity. In environments built on Apache Cassandra, for example, policies can automatically govern replication strategies by enforcing placement rules across data centers and geographic regions. Such policies ensure that availability and latency objectives are met while adhering to data residency constraints. Similarly, in MongoDB-based platforms, retention and access policies can be applied at the collection or document level, enabling automated enforcement of lifecycle and privacy requirements.

For DataStax Enterprise deployments, policy rules can trigger archival workflows that migrate cold or infrequently accessed data to lower-cost storage tiers. These actions can be executed transparently, without requiring application developers to implement custom governance logic. By externalizing such controls, enterprises reduce the risk of inconsistent behavior across applications. The policy-driven approach also simplifies operations by replacing brittle scripts with reusable, centrally managed rules.

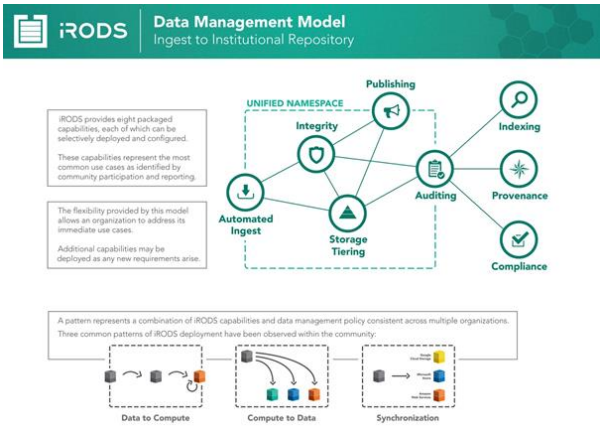

Figure 1. iRODS Policy-Driven Data Management Architecture

Embedding policy evaluation directly within the data management layer yields significant benefits in terms of consistency, scalability, and maintainability. Rather than relying on each application to correctly interpret and enforce governance requirements, policy-driven architectures provide a shared enforcement mechanism that applies uniformly across the platform. This reduces configuration drift and minimizes the likelihood of human error in complex operational environments. Centralized policy management also improves visibility, enabling administrators to understand which rules are in effect and how they influence system behavior. From an organizational perspective, this approach supports clearer separation of responsibilities between governance, operations, and application development teams. Governance teams can define and update policies, while platform teams focus on ensuring reliable execution. As enterprise data ecosystems continue to grow in size and heterogeneity, embedding policy-driven automation at the data management layer becomes a critical enabler of sustainable, compliant, and resilient Big Data operations.

Declarative Policy Evaluation and Access Control

Figure 2 presents the XACML reference architecture, which formalizes policy-driven authorization through a clear separation of concerns across four well-defined components: the Policy Administration Point (PAP), Policy Decision Point (PDP), Policy Enforcement Point (PEP), and Policy Information Point (PIP). The PAP is responsible for authoring and managing policies, allowing governance teams to express access and usage requirements in a standardized, declarative form. The PDP evaluates incoming access requests against these policies, taking into account contextual information and applicable rules. The PEP acts as the gatekeeper, intercepting requests and enforcing the decisions returned by the PDP. The

© 2019 IJSRET

PIP supplies additional attributes, such as user roles, data classifications, or environmental conditions, that are required for accurate policy evaluation. Together, these components form a robust and extensible framework for policy-driven authorization. Although originally designed for access control, the architectural principles embodied in XACML generalize to broader governance and automation scenarios. This makes the model particularly valuable as a reference for enterprise data platform design.

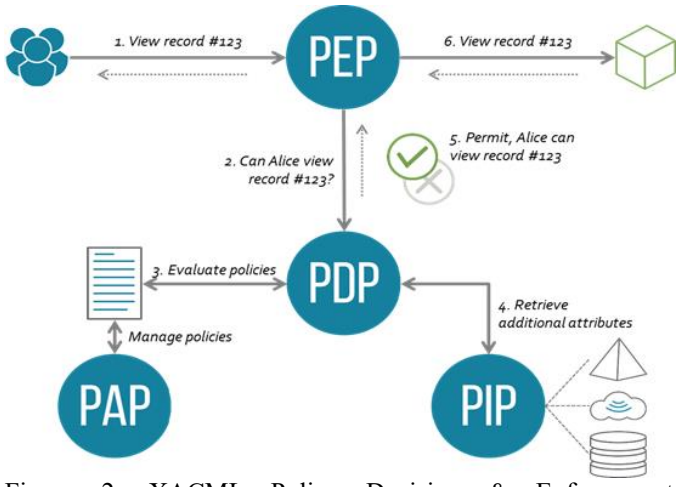

Figure 2. XACML Policy Decision & Enforcement Architecture

In distributed databases such as MongoDB and Apache Cassandra, access control requirements have evolved beyond simple role-based permission models. Modern enterprise deployments often require policies that account for user attributes, data sensitivity levels, geographic location, and operational context. For example, access to personally identifiable information may depend not only on a user's role but also on the purpose of access, time of day, or regulatory jurisdiction. XACML-style architectures enable these attribute-based and context-aware policies to be evaluated dynamically at runtime, rather than being hard-coded into application logic. This dynamic evaluation allows policies to adapt as conditions change, without requiring redeployment of applications or database services. In MongoDB, such models can complement native role-based controls by introducing higher-level decision logic. In Cassandra-based systems, external policy enforcement layers can mediate access to distributed data services. The result is a more flexible and expressive access control framework aligned with enterprise governance needs.

The ability to incorporate rich contextual information into authorization decisions is particularly important in regulated industries such as finance, healthcare, and telecommunications. Regulatory frameworks often require organizations to demonstrate not only that access is restricted, but also that it is appropriate for a given context and purpose. XACML-style architectures support this requirement by providing explicit decision points where policies can be evaluated and logged. These evaluation traces create an auditable record of why a particular access decision was made, supporting compliance reporting and forensic analysis. Additionally, the separation of policy administration from enforcement enables organizations to respond rapidly to regulatory changes by updating policies centrally. This reduces the risk of inconsistent enforcement across distributed systems. By extending XACML principles beyond traditional authorization into broader data governance workflows, enterprises can build policy-driven platforms that are both secure and compliant at scale.

Operational Automation Through Policy-Triggered Workflows

While policy engines determine what actions should occur in response to defined rules and constraints, workflow orchestration systems operationalize how those actions are executed across distributed infrastructure. Figure 3 illustrates the Directed Acyclic Graph (DAG) model used by Apache Airflow, in which complex processes are decomposed into discrete, interdependent tasks with explicit execution order and dependencies. This model provides a clear and reproducible representation of operational logic, enabling workflows to be versioned, monitored, and replayed as needed. Each task within a DAG can represent a specific enforcement or remediation action, such as validating metadata, moving data between storage tiers, or updating configuration state. By integrating with policy engines, orchestration systems translate abstract governance decisions into concrete operational steps. This separation allows workflows to evolve independently of policy definitions. As a result, enterprises can refine execution strategies without altering the underlying governance intent. The DAG abstraction also supports parallelism and fault tolerance, which are essential for large-scale data platforms. Together, policy engines and workflow orchestrators form a cohesive automation layer that bridges intent and execution.

© 2019 IJSRET

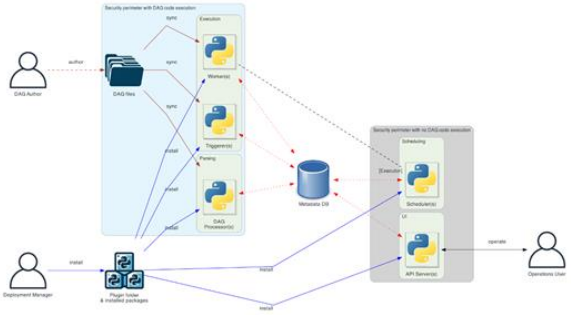

Figure 3. Apache Airflow Workflow (DAG-Based Automation Model)

In enterprise data platforms, policy-triggered workflows are commonly used to automate a wide range of governance and operational tasks that would otherwise require manual intervention. Schema validation workflows can be triggered when new datasets are ingested, ensuring that structural and semantic constraints are enforced consistently. Data quality checks can be executed automatically to detect anomalies, missing values, or constraint violations before data is made available to downstream consumers. Backup verification workflows help ensure that recovery mechanisms function as intended, reducing the risk of data loss in the event of failures. Compliance reporting tasks can be scheduled or triggered in response to policy evaluations, generating evidence required for audits and regulatory reviews. For example, in an Apache Cassandra cluster, a policy violation such as unauthorized replication outside an approved geographic region can be detected by a policy engine. This event can trigger an Airflow DAG that remediates the configuration, records the incident, and notifies relevant stakeholders. Such automated responses reduce mean time to resolution and improve overall governance reliability.

The integration of policy-driven decision-making with workflow orchestration aligns closely with DataOps practices, which emphasize automation, observability, and continuous improvement across the data lifecycle. By embedding policy-triggered workflows into operational pipelines, organizations can enforce governance continuously rather than relying on periodic manual reviews. This approach enables early detection of violations and rapid remediation, minimizing business and compliance risk. Workflow execution logs and metadata also provide valuable visibility into how policies are enforced in practice, supporting auditing and root-cause analysis. Moreover, orchestrated workflows can be extended to incorporate approval steps, escalation paths, or human-in-the-loop interventions when required. As enterprise data ecosystems grow in complexity, the combination of policy engines and workflow orchestration offers a scalable and adaptable mechanism for maintaining control. Ultimately, this integration enables enterprises to operationalize governance as code, embedding compliance and reliability directly into the fabric of their data platforms.

Policy-Driven Automation in Big Data Platforms

Modern Big Data technologies increasingly expose extensibility points and management interfaces that make them amenable to policy integration. MongoDB, for instance, provides fine-grained role-based access control mechanisms that can be augmented with attribute-aware decision logic, enabling organizations to express nuanced access policies aligned with data sensitivity and user context. Apache Cassandra and DataStax Enterprise similarly expose configuration controls for replication, consistency levels, encryption, and authentication, allowing governance requirements to be enforced at the infrastructure level. These native capabilities form a foundation upon which external policy engines can operate, evaluating system state and triggering automated actions when policy conditions are met. By interfacing with management APIs, configuration services, or metadata catalogs, policy engines can observe and influence platform behavior without invasive changes to core systems. When combined with workflow orchestration tools, these integrations enable end-to-end automation that spans policy evaluation, remediation, and auditing. The result is a cohesive automation layer that unifies governance across heterogeneous Big Data platforms.

Policy-driven automation is particularly valuable in multi-cluster and hybrid environments, where enterprise data is distributed across on-premise infrastructure and multiple cloud providers. In such settings, ensuring consistent governance is challenging due to differences in platform capabilities, deployment models, and operational practices. Declarative policies provide a unifying abstraction by expressing governance requirements independently of where data resides or how it is physically stored. For example, a single policy can specify replication constraints or access restrictions that apply uniformly across Cassandra clusters in different regions and MongoDB deployments spanning on-premise and cloud environments. External policy engines interpret these rules and translate them into platform-specific enforcement actions. This approach reduces duplication of governance logic and minimizes the risk of inconsistent enforcement. It also simplifies the introduction of new environments, as existing policies can be reused without extensive reconfiguration. As

© 2019 IJSRET

enterprises increasingly adopt hybrid architectures, policy-driven automation becomes a critical enabler of operational coherence.

Declarative policies also play a central role in encoding and enforcing compliance obligations that arise from regulatory and organizational requirements. Policies can specify retention limits, ensuring that data is automatically archived or deleted after defined periods. Geographic data residency rules can be enforced by restricting replication or access to approved regions, supporting compliance with jurisdictional regulations. Because these policies are defined once and applied centrally, they provide a clear and auditable representation of compliance intent. Automated enforcement reduces reliance on manual checks and periodic audits, lowering the likelihood of violations. Moreover, centralized policy management enables rapid adaptation to regulatory change, as updates can be propagated across platforms without modifying application code. In this way, policy-driven automation transforms compliance from a reactive, manual process into a proactive, systematic capability embedded within enterprise Big Data platforms.

Key Studies and Influential Work

Several foundational studies underpin the feasibility and effectiveness of policy-driven automation in enterprise data platforms, establishing both the theoretical and practical basis for its adoption. Early research on policy-driven dataset management demonstrated that rules and declarative policies could be used to unify storage management, metadata handling, and data lifecycle control within distributed systems. These studies showed that administrative intent could be expressed independently of specific storage technologies, enabling systems to adapt dynamically to changing workloads and operational conditions.

By abstracting governance concerns into policies, researchers illustrated how complex management tasks could be automated in a consistent and repeatable manner. This work laid the groundwork for treating data management as a programmable discipline rather than a collection of manual procedures. Importantly, these early efforts addressed core challenges such as scalability, fault tolerance, and long-term preservation. The results provided empirical evidence that policy-based approaches could reduce operational overhead while improving system reliability. As distributed data systems became more prevalent, these insights gained increasing relevance. They continue to inform contemporary designs for automated data governance.

The iRODS project represents a significant milestone in the evolution of policy-driven data management, providing one of the first production-grade implementations of rule-oriented automation at scale. By embedding a rule engine directly into the data management layer, iRODS demonstrated how policies could govern data placement, replication, access, and preservation across heterogeneous storage resources. Its use of microservices to encapsulate enforcement logic allowed policies to trigger concrete actions without tight coupling to underlying infrastructure. This architectural approach influenced subsequent enterprise platforms by showing that policy-driven automation could operate reliably in real-world environments. iRODS also emphasized the importance of metadata as a first-class input to policy evaluation, enabling more informed and context-aware decisions. Over time, the project accumulated practical experience from deployments in scientific, governmental, and enterprise settings. These experiences validated the scalability and robustness of rule-oriented management. As a result, iRODS became a reference point for both academic research and industrial implementations of policy-driven data platforms.

Standardization efforts such as the development of XACML further strengthened the case for policy-driven automation by formalizing common architectures and interfaces for policy evaluation and enforcement. By defining clear roles for policy administration, decision-making, information retrieval, and enforcement, XACML provided a reusable blueprint for building governance systems that could span multiple technologies and organizational boundaries. In parallel, the emergence of DataOps and workflow orchestration tools highlighted the operational dimension of policy enforcement, emphasizing the need to automate governance alongside data processing pipelines. These tools demonstrated how policy decisions could be translated into repeatable, observable workflows that integrate seamlessly with day-to-day operations. Together, these strands of work bridged the gap between high-level governance intent and low-level execution. They showed that policy-driven automation is not only theoretically sound but also operationally practical. Collectively, these studies validate policy-driven automation as a viable and scalable approach for managing large-scale, heterogeneous enterprise data platforms.

Case Study: Policy-Driven Automation in a Hybrid Enterprise Data Platform

A large financial services enterprise operating across multiple geographic regions provides a representative case study for policy-driven automation in practice. The organization maintained a hybrid data platform composed of on-premise infrastructure and public cloud deployments, using MongoDB

© 2019 IJSRET

for customer-facing transactional workloads and Apache Cassandra (via DataStax Enterprise) for high-throughput event and ledger data. Rapid growth in data volume, coupled with stringent regulatory requirements related to data residency, retention, and access auditing, exposed the limitations of manual governance processes. Configuration drift across clusters, inconsistent access controls, and delayed compliance reporting created operational risk and increased audit overhead. To address these challenges, the organization adopted a policy-driven automation approach that externalized governance logic from applications and platform-specific scripts.

At the core of the solution was a centralized policy layer responsible for defining declarative rules governing access control, replication boundaries, and data lifecycle management. Policies specified constraints such as "customer data originating in Region A must not be replicated outside approved jurisdictions" and "personally identifiable information must be retained for no longer than a fixed regulatory period." These policies were evaluated continuously against platform state using metadata collected from MongoDB and Cassandra clusters. When violations were detected such as unauthorized replication or expired retention windows the policy engine generated enforcement decisions without human intervention. This decoupling allowed governance teams to update compliance rules centrally in response to regulatory changes, without requiring application redeployments or platform reconfiguration by hand.

Workflow orchestration played a critical role in translating policy decisions into operational outcomes. Policy violations triggered automated workflows that executed corrective actions, such as reconfiguring Cassandra replication strategies, revoking access privileges in MongoDB, or initiating secure data deletion and archival processes. These workflows also generated audit artifacts, including decision logs and remediation records, which were stored centrally for compliance review. By automating remediation and evidence collection, the organization significantly reduced mean time to resolution for governance issues and improved audit readiness. Importantly, the workflows were designed to support human-in-the-loop approval for high-risk actions, balancing automation with operational oversight.

The outcomes of the initiative demonstrated the practical value of policy-driven automation at scale. The organization achieved consistent governance enforcement across heterogeneous data platforms and deployment environments, reduced manual operational effort, and improved compliance transparency. Configuration errors and policy violations were detected and resolved earlier in the data lifecycle, lowering regulatory and reputational risk. The case study illustrates how policy-driven automation can transform enterprise data governance from a reactive, labor-intensive function into a proactive, scalable capability. It also highlights the broader applicability of policy-driven architectures for enterprises seeking to manage complex Big Data platforms while maintaining agility, resilience, and regulatory compliance.

III. CONCLUSION

Policy-driven automation represents a critical evolution in enterprise data platform management, addressing the growing gap between system scale and human operational capacity. By explicitly separating governance intent from execution logic, organizations gain the ability to express complex rules and constraints at a high level while relying on automated systems to enforce them consistently. This separation reduces the fragility associated with hard-coded controls and manual procedures, which often fail to keep pace with dynamic workloads and infrastructure changes. Governance policies become living artifacts that can be reviewed, tested, and evolved independently of application code. As a result, enterprises can respond more rapidly to regulatory updates, security threats, and business priorities. Policy-driven approaches also improve transparency by making governance intent explicit and machine-readable. This clarity supports stronger collaboration between governance, security, and engineering teams. Ultimately, policy-driven automation transforms governance from an after-the-fact oversight function into an integral part of platform design. It establishes a foundation for scalable, trustworthy data operations in complex enterprise environments.

As Big Data platforms such as MongoDB, Apache Cassandra, and DataStax Enterprise continue to underpin enterprise analytics and transactional workloads, the need for automated governance becomes increasingly pronounced. These platforms are designed for horizontal scalability and high availability, often spanning multiple clusters, regions, and deployment models. While such architectures deliver performance and resilience benefits, they also magnify the consequences of misconfiguration and inconsistent policy enforcement. Policy-driven architectures help mitigate these risks by providing a uniform governance layer that operates across diverse technologies. Access controls, replication strategies, and data lifecycle rules can be defined once and applied consistently, regardless of where data resides. This uniformity reduces operational complexity and minimizes the likelihood of compliance gaps. Additionally, policy-driven automation enables proactive detection and remediation of

© 2019 IJSRET

issues before they escalate into incidents. In this way, governance becomes a continuous, embedded capability rather than a periodic, manual exercise.

Looking forward, policy-driven architectures will play an increasingly central role in enabling resilient, compliant, and adaptive data ecosystems. As enterprises adopt hybrid and multi-cloud strategies, the ability to enforce governance consistently across organizational and technological boundaries will be essential. Policy-driven automation provides the abstraction necessary to manage this complexity while preserving flexibility and performance. Advances in metadata management, observability, and workflow orchestration will further enhance the effectiveness of policy-based systems. These developments will enable richer context-aware policies and more sophisticated automated responses. At the same time, human oversight will remain important for defining intent and managing exceptions. By combining declarative governance with automated execution, organizations can build data platforms that are both agile and trustworthy. In doing so, they position themselves to leverage data at scale without sacrificing security, compliance, or operational stability.

REFERENCES

1. Robinson, W. N., & Pawlowski, S. D. (2002). Managing requirements inconsistency with development goal monitors. IEEE Transactions on Software Engineering, 25(6), 816-835. https://doi.org/10.1109/32.824411
2. Chaudhuri, S., Narasayya, v. (2007). Self-tuning database systems: A decade of progress. Proceedings of the VLDB Endowment, 1(1), 3–14. https://dl.acm.org/doi/10.5555/1325851.1325856
3. Stonebraker, M., Çetintemel, U., & Zdonik, S. (2005). The 8 requirements of real-time stream processing. ACM SIGMOD Record, 34(4), 42–47. https://doi.org/10.1145/1107499.1107504
4. Buneman, P., Khanna, S., & Wang-Chiew, T. (2001). Why and where: A characterization of data provenance. In International conference on database theory (pp. 316-330). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-44503-X_20
5. Rajasekar, A., Moore, P., Hou, C. Y., Lee, C. A., Marciano, R., de Torcy, A., & Wan, M., Schroeder, W., Chen, S.Y., Gilbert, L., Tooby, P., Zhu, B. (2010). iRODS primer: Integrated rule-oriented data system. Synthesis Lectures on Information Concepts, Retrieval, and Services, 2(1), 1–143. https://dl.acm.org/doi/10.5555/1855046
6. Simmhan, Y. L., Plale, B., & Gannon, D. (2005). A survey of data provenance in e-science. ACM Sigmod Record, 34(3), 31-36. https://doi.org/10.1145/1084805.1084812
7. Hu, V. C., Kuhn, D. R., Ferraiolo, D. F., & Voas, J. (2015). Attribute-based access control. Computer, 48(2), 85-88. https://doi.org/10.1109/MC.2015.33
8. Ferraiolo, D. F., Sandhu, R., Gavrila, S., Kuhn, D. R., & Chandramouli, R. (2001). Proposed NIST standard for role-based access control. ACM Transactions on Information and System Security (TISSEC), 4(3), 224-274. https://doi.org/10.1145/501978.501980
9. Ganapathi, A., Kuno, H., Dayal, U., Wiener, J. L., Fox, A., Jordan, M., & Patterson, D. (2009). Predicting multiple metrics for queries: Better decisions enabled by machine learning. IEEE International Conference on Data Engineering. pp. 592-603. https://doi.org/10.1109/ICDE.2009.130
10. Dean, J., & Ghemawat, S. (2004). MapReduce: Simplified data processing on large clusters. OSDI. Volume 6, Page 10. https://dl.acm.org/doi/10.5555/1251254.1251264
11. Kleppmann, M. (2017). Designing data-intensive applications. https://www.oreilly.com/library/view/designing-data-intensive-applications/9781491903063/
12. Zhu, X., Young, D., Watson, B. J., Wang, Z., Rolia, J., Singhal, S. et al. (2009). 1000 islands: an integrated approach to resource management for virtualized data centers. Cluster Computing, 12(1), 45-57. https://doi.org/10.1007/s10586-008-0067-6
13. Burgess, M. (1995). A site configuration engine. Computing systems, 1-29. https://markburgess.org/papers/paper1.pdf
14. Morris, K. (2016). Infrastructure as code: managing servers in the cloud. " O'Reilly Media, Inc.". https://www.oreilly.com/library/view/infrastructure-as-code/9781491924334/
15. Lakshman, A., & Malik, P. (2010). Cassandra: A decentralized structured storage system. ACM SIGOPS Operating Systems Review, 44(2), 35–40. https://doi.org/10.1145/1773912.1773922
16. Otto, B. (2011). Organizing Data Governance: Findings from the Telecommunications Industry and Consequences for Large Service Providers. Communications of the Association for Information Systems, 29, pp-pp. https://doi.org/10.17705/1CAIS.02903
17. Khatri, V., & Brown, C. V. (2010). Designing data governance. Communications of the ACM, 53(1), 148-152. https://doi.org/10.1145/1629175.1629210
18. Cooper, B. F., Cooper, B. F., Ramakrishnan, R. Bohannon, P. Puz, N. et al. (2008). PNUTS: Yahoo!'s hosted data serving platform. Proceedings of the VLDB Endowment,

© 2019 IJSRET

1(2), 1277–1288. https://doi.org/10.14778/1454159.1454167
19. Abadi, D. (2012). Consistency tradeoffs in modern distributed database system design. IEEE Computer, 45(2), 37–42. https://doi.org/10.1109/MC.2012.33
20. Sudhir Vishnubhatla. (2019). From Rules To Neural Pipelines: NLP-Powered Automation For Regulatory Document Classification In Financial Systems. In International Journal of Science, Engineering and Technology (Vol. 7, Number 1). Zenodo. https://doi.org/10.5281/zenodo.17473977.
21. White, S. R., Hanson, J. E., Whalley, I., Chess, D. M., & Kephart, J. O. (2004). An architectural approach to autonomic computing. In International Conference on Autonomic Computing, 2004. Proceedings. (pp. 2-9). IEEE. https://doi.org/10.1109/ICAC.2004.1301340
22. Weber, K., Otto, B., & Österle, H. (2009). One size does not fit all---a contingency approach to data governance. Journal of Data and Information Quality (JDIQ), 1(1), 1-27. https://doi.org/10.1145/1515693.1515696
23. Kephart, J. O., & Chess, D. M. (2003). The vision of autonomic computing. IEEE Computer, 36(1), 41–50. https://doi.org/10.1109/MC.2003.1160055
24. Missier, P., Belhajjame, K., & Cheney, J. (2013, March). The W3C PROV family of specifications for modelling provenance metadata. In Proceedings of the 16th international conference on extending database technology (pp. 773-776). https://doi.org/10.1145/2452376.2452478
25. Calero, J. M. A., Edwards, N., Kirschnick, J., Wilcock, L., & Wray, M. (2010). Toward a multi-tenancy authorization system for cloud services. IEEE Security & Privacy, 8(6), 48–55. https://doi.org/10.1109/MSP.2010.194

© 2019 IJSRET