Claims

1. **A generative DevOps automation system**, comprising:
2. one or more **generative AI micro-agents** deployed entirely within a user's cloud account environment, each micro-agent configured to interact with cloud infrastructure resources and associated with an identity having **least-privilege access permissions** to perform a restricted set of tasks;
3. a **reinforcement learning optimization engine** communicatively coupled to the micro-agents, the optimization engine configured to evaluate proposed actions generated by the micro-agents according to a multi-objective reward function that balances **(i)** predicted cost impact, **(ii)** predicted performance impact, and **(iii)** estimated compliance or security risk of the actions, and to select or recommend actions for execution based on an overall optimized reward;
4. a dynamic **policy enforcement layer** (Policy Weaver) integrated with the optimization engine, the policy enforcement layer comprising a set of machine-readable **guardrail policies** and a real-time evaluator that analyzes each proposed infrastructure change or cloud action **before deployment** to determine compliance with the guardrail policies, wherein if a proposed change violates a policy or would cause a violation, the policy enforcement layer automatically blocks the change or modifies it to enforce compliance; and
5. a **federated learning coordinator** configured to facilitate continual learning across a plurality of separate deployments of the system, by aggregating model parameter updates or gradients from each deployment that are **transmitted in encrypted or privacy-preserving form**, computing an updated set of model parameters based on the aggregated updates, and distributing the updated model parameters back to the micro-agents or optimization engines in the respective deployments, thereby improving the agents' performance without requiring sharing of sensitive raw data between the deployments.

6. The system of claim 1, wherein each generative AI micro-agent comprises a specialized autonomous agent for a particular DevOps domain or task, and wherein the micro-agent is powered by a **foundation model** (or large language model) accessed via a **private endpoint** within the user's cloud account such that no agent prompts or data are sent over a public internet connection[9].

7. The system of claim 1, wherein the identity associated with each micro-agent is an **isolated IAM role or cloud identity** granting only a minimal set of permissions needed for that micro-agent's function, and wherein any attempt by the micro-agent to perform an action outside its permissions is denied by the cloud platform's inherent access controls, thereby containing the agent's potential impact[11][12].

8. The system of claim 1, wherein the reinforcement learning optimization engine implements a **constrained multi-objective RL policy** that treats compliance violations as constrained conditions to avoid, such that the engine will not select any action whose estimated compliance risk exceeds a defined threshold (thereby effectively treating compliance as a hard constraint while optimizing cost and performance objectives)[6][18].

9. The system of claim 1, wherein the **estimated compliance risk** for a proposed action is determined by the policy enforcement layer or a predictive model trained on historical policy violations, and wherein the optimization engine assigns a **penalty in the reward** for actions with higher compliance risk, disincentivizing the micro-agents from proposing non-compliant actions.

10. The system of claim 1, wherein the guardrail policies in the policy enforcement layer include one or more of:
    a) **Identity and access policies** that prevent creation of cloud roles, credentials, or permissions that exceed pre-defined least privilege limits (including detection of wildcard or overly broad permissions)[21][11];
    b) **Resource configuration policies** that enforce security best practices and tagging requirements on cloud resources (including requiring encryption on storage resources[3], mandatory tags such as "Owner" or "CostCenter" on resources[3], network restrictions on public exposure, etc.);
    c) **Cost/budget policies** that prohibit or warn against actions breaching budget thresholds or deploying resource types disallowed for cost reasons;
    d) **Regulatory compliance policies** and **service control policies** that restrict actions violating external regulations or organizational rules (including geographic restrictions, data residency requirements, usage of approved services only, etc.); and
    e) **Infrastructure drift or consistency policies** that ensure AI-generated changes do not conflict with existing infrastructure state or cause out-of-process configuration drift.

11. The system of claim 6, wherein the policy enforcement layer is further configured to **auto-remediate** certain violations by modifying the micro-agent's proposed code or parameters (including adding missing tags, enabling encryption flags, narrowing IAM permissions), and to **provide feedback** of these modifications to the generative micro-agent or the optimization engine, such that future proposals inherently comply with said policies (realizing a feedback-based learning of organizational requirements by the AI agent).

12. The system of claim 1, wherein the policy enforcement layer interfaces with external policy-as-code engines or scanning tools as plug-ins (for example, using an Open Policy Agent evaluator, or invoking static analysis tools on generated code) to evaluate compliance of proposed changes, thereby allowing **extensible policy checks** without modifying the core system.

13. The system of claim 1, wherein the federated learning coordinator utilizes a **secure aggregation protocol** or **differential privacy mechanisms** to ensure that individual deployments' model updates do not reveal proprietary information, such that agents share

only “sanitized” gradients or encrypted parameter deltas[13] and the coordinator only produces a combined result that cannot be reverse-engineered to individual inputs[14].

14. The system of claim 1, wherein the federated learning coordinator is further configured to manage a **global model version**, optionally performing validation on aggregated updates (including evaluating the updated model on a set of tests or rules) before releasing the update to all micro-agents, thereby preventing the propagation of a faulty or non-compliant model update across deployments.

15. The system of claim 1, further comprising a **logging and audit subsystem** that records each action taken or recommended by a micro-agent, each decision by the RL optimization engine, and each intervention by the policy enforcement layer, along with timestamps and reasoning metadata, whereby an administrator can review a complete, time-sequenced audit trail of the system’s autonomous operations for compliance and governance purposes.

16. The system of claim 1, wherein the generative AI micro-agents, the RL optimization engine, and the policy enforcement layer are implemented as software components executing on cloud-based computing instances or serverless functions within the user’s account, and wherein communications between these components and with cloud provider APIs are restricted to **private network endpoints** or secure channels, thereby ensuring all operational data and instructions remain within controlled network boundaries[9][10].

17. A **method for governance-aware automated cloud operations**, comprising:
(a) deploying, within a cloud account of a user, one or more AI-based micro-agents each having a limited-access identity and configured to generate proposals for cloud infrastructure changes or DevOps actions in response to observed conditions or received objectives;
(b) for each proposal generated by a micro-agent, evaluating the proposal using a **reinforcement learning agent** that computes a reward with multiple components reflecting at least cost efficiency, performance improvement, and compliance risk associated with the proposal, and selecting whether to approve, adjust, or reject the proposal based on the multi-component reward (wherein proposals with unacceptable compliance risk contribute negative reward and are disfavored in future agent policy);
(c) subjecting approved proposals to a **real-time policy compliance check** prior to deployment, by programmatically analyzing the proposal against a set of predefined guardrail rules that encode security, compliance, and governance constraints, and upon detecting any violation or deviation: either (i) automatically modifying the proposal to correct the violation, or (ii) blocking the execution of the proposal and logging a policy exception event;
(d) if the proposal is cleared (either originally or after modification) by the compliance check, executing the proposal to effect changes in the cloud account (thereby implementing the agent’s action in a compliant manner), and recording the outcome and any metrics resulting from the change; and
(e) providing **feedback to the AI micro-agents and the RL agent** based on the outcome and compliance check results, such that the micro-agents update their generative behavior

(for example, via adjusted model parameters or prompt conditioning) and the RL agent updates its policy to increasingly favor actions that succeed (in performance and cost terms) and comply with all policies, thereby continuously learning optimal actions under the user's specific operational constraints.

18. The method of claim 13, wherein step (c) comprises parsing an **infrastructure-as-code (IaC) template or script** generated by a micro-agent (such as a Terraform, CloudFormation, or Kubernetes configuration file) and evaluating the parsed representation with policy rules that detect security misconfigurations or compliance issues in the IaC prior to its application, effectively acting as an automated code review and enforcement gate in the deployment pipeline[27][20].

19. The method of claim 13, wherein the reinforcement learning agent in step (b) uses a **neural network policy** trained via deep reinforcement learning, and the method further comprises an initial training phase wherein the agent is trained in a simulated cloud environment or using historical operations data to accelerate learning of the multi-objective policy prior to deployment in the live cloud account.

20. The method of claim 13, further comprising: **(f)** deploying the system in a plurality of different cloud accounts or environments (which may belong to one organization or multiple organizations), each deployment operating its local micro-agents and RL evaluations on local data; **(g)** periodically aggregating knowledge across said deployments by collecting model updates from each deployment's AI components and performing a **federated learning round** to compute a refined global model; and **(h)** distributing updated model parameters or policies back to each deployment, thereby enabling each deployment's AI to benefit from learnings across all deployments without exchanging raw operational data.

21. The method of claim 16, wherein collecting model updates comprises receiving, from each deployment, an **encrypted or obfuscated set of model weight updates** such that the method preserves privacy – for example, each deployment shares only noised gradients or homomorphically encrypted parameters – and wherein performing the federated learning round comprises combining the updates (e.g., averaging) to produce a global update, and wherein distributing updated parameters comprises sending the combined update to each deployment which then applies it to its local model after verifying the update's integrity.

22. The method of claim 13, wherein the micro-agents are configured to incorporate **feedback loops with human operators** such that when the policy enforcement layer blocks a proposed action, the system can notify a human (via a dashboard or message) with an explanation of the violation, receive a revised instruction or override from the human, and feed that back into the agent's plan – thereby allowing human-in-the-loop control for exceptional situations while generally automating compliant behavior.

23. The method of claim 13, further comprising **retraining the generative model or agent models in-account using accumulated data**, including: gathering logs of agent interactions, states, and decisions; fine-tuning the generative AI model on examples of successful (compliant, efficient) actions vs. unsuccessful ones; and updating the RL

policy using off-policy learning or batch reinforcement learning on the logged data, all performed within the secure boundaries of the user's account so that proprietary data is not exposed externally.

24. The method of claim 13, wherein the guardrail rules used in step (c) are **stored and managed as version-controlled code** (policy-as-code), enabling the organization to update governance rules (e.g., add a new security requirement) and have the updated rules automatically applied to all new proposals by the AI without modifying the AI's code, and wherein the method automatically tests proposals against the updated rules (including possibly re-evaluating past proposals or deployments for compliance drift).

25. The method of claim 13, wherein the cloud account environment is part of a **multi-cloud or hybrid cloud** infrastructure, and the method further comprises: deploying at least one micro-agent or a subset of system components in a first environment (e.g., an on-premises data center or a second cloud provider) and other components in a second environment (e.g., a public cloud account), and coordinating actions between them such that the policy enforcement layer enforces a unified set of guardrails across both environments.

26. The method of claim 21, wherein communications between the first and second environment are secured via end-to-end encryption and the system ensures that any action affecting the first environment's resources is approved by the policy logic applicable to that environment, thereby extending the governance-aware AI automation across hybrid deployments.

27. A **computer-readable non-transitory storage medium** containing instructions that, when executed by one or more processors, configure a system to perform the steps of the method of claim 13, thereby implementing an in-account, governance-aware generative DevOps AI system as a software product.

28. A **computer-readable non-transitory storage medium** containing instructions that, when executed, instantiate: a set of generative AI micro-agents with associated least-privilege credentials for cloud APIs; a reinforcement learning decision module that guides the micro-agents based on multi-objective optimization of cost, performance, and compliance; a policy enforcement engine that intercepts and validates agent-generated cloud configurations against compliance rules prior to deployment; and a federated learning client that enables the system to partake in cross-account model training – wherein the integrated operation of these instantiated components achieves autonomous cloud management that adheres to organizational guardrails.

29. **An extensible AI DevOps platform**, comprising:

30. a **core orchestration framework** configured to manage AI-driven automation workflows for cloud infrastructure provisioning and operations;
31. a **plug-in interface** defining one or more extension points for integrating additional modules into the platform, including **agent plug-ins** that provide new generative micro-agent capabilities and **policy plug-ins** that provide additional guardrail rules or compliance checks;

32. wherein the core orchestration framework includes a governance enforcement subsystem that ensures any actions taken by an agent plug-in are subject to the policy plug-ins' rules, thereby maintaining compliance even as the platform is extended; and
33. wherein new plug-in modules can be added or updated without interrupting the running system, allowing the platform to adapt to new cloud services, custom organizational policies, or updated AI models while still operating under the same reinforcement learning and Policy Weaver governance mechanisms.

34. The platform of claim 25, wherein the plug-in interface for agent modules requires each agent plug-in to specify its necessary permissions and scope of actions, and upon installation of an agent plug-in, the platform automatically generates or configures a **least-privileged execution role** for that agent, so that the agent plug-in runs within a sandbox of allowed actions.

35. The platform of claim 25, wherein the **policy plug-ins** are provided as code artifacts (scripts, rules, or machine learning models) that the platform loads into the Policy Weaver engine, and the platform supports versioning and conditional activation of policy plug-ins (for example, enabling certain compliance rules only in specific accounts or regions), thereby offering flexible governance control through modular policy deployment.

36. The platform of claim 25, further comprising a **retraining orchestrator** that is also modular – including support for **model training plug-ins** that implement custom training logic or integration with external ML pipelines – so that the platform can incorporate new learning techniques or retraining workflows (such as a plug-in for federated multi-task learning or a plug-in that triggers model retraining based on a schedule or drift detection), all under the umbrella of the platform's security and compliance controls.

37. **A method of improving a generative DevOps AI model via federated training**, comprising: deploying instances of a governance-aware DevOps AI system according to claim 1 in multiple distinct cloud environments; logging, at each instance, data about the instance's local state, actions taken by AI agents, and outcomes including any compliance rule evaluations; periodically, at each instance, computing an **incremental model update** (for at least one of: the generative foundation model used by the agents, or the reinforcement learning policy model) based on the logged data; transmitting the incremental model updates from the instances to a central aggregation service, with each update **encrypted or obfuscated** such that the central service cannot discern proprietary details of any single instance's data; at the central service, combining the updates to produce an **aggregated global model update**; and sending the aggregated global model update back to each instance for incorporation into its local models, thereby synchronizing the model's improvements across all instances; **wherein** each instance continues operating after incorporating the update, resulting in progressively improved performance and policy compliance of the AI agents across all environments due to the collective learning.

38. The method of claim 29, wherein the aggregation service performs a secure multi-party computation protocol involving the instances such that no single point (neither the instances nor the service alone) has access to the raw model updates in unencrypted form,

and only the final aggregated result is decrypted for use, thus preserving confidentiality of each instance's training data throughout the federated learning process[13].

39. The method of claim 29, further comprising evaluating the aggregated global model update on a **validation dataset or ruleset** that represents essential safety and compliance behaviors (for example, tests ensuring that the updated model does not suggest actions that violate critical policies), and conditioning the acceptance of the global update on passing these evaluations, such that model changes that would degrade compliance are not applied.

40. **A hybrid deployment system** for generative AI-driven cloud operations, comprising: a **first computing environment** (on-premises or in a secure private cloud) hosting a centralized governance service including the Policy Weaver and optionally the RL optimization engine; and a **second computing environment** (a public cloud account) hosting one or more micro-agents and an interface to a generative AI model; wherein the micro-agents in the second environment propose infrastructure changes which are communicated to the first environment's governance service for compliance checking and approval, and only upon receiving approval does the second environment execute the changes; **whereby** the sensitive compliance enforcement logic and audit trail are maintained in the first environment, while the scalability and functionality of cloud-hosted generative AI are utilized in the second environment, achieving a secure **hybrid architecture**.

41. The system of claim 32, wherein the communication between the micro-agents and the centralized governance service is implemented via a secure API or messaging queue with authentication, such that no unauthorized agent or system can bypass the governance checks; and wherein the centralized governance service can manage multiple cloud accounts (acting as a hub for governance across distributed AI agents), providing a unified policy control point for multi-account or multi-cloud AI operations.

42. The system of claim 32, wherein the RL optimization engine is split between environments such that a local component in the second environment observes real-time metrics and suggests candidate actions, while a global component in the first environment (having knowledge of organizational priorities and constraints) refines or selects the final action from the candidates, thus partitioning the decision process between a cloud-side part and a governance-side part.

43. The system of claim 1, wherein the reinforcement learning optimization engine and policy enforcement layer are together configured to enable the AI system to **operate within regulated industries** by ensuring that any action that could impact regulatory compliance (for example, moving data to a different jurisdiction, altering a security control, or affecting audit settings) is identified and subjected to enhanced scrutiny or additional approval steps.

44. The system of claim 1, wherein the micro-agents include an agent specifically for **monitoring and mitigating AI behavior**, which uses a machine learning model to watch the other micro-agents' proposals and flag potentially risky or anomalous behaviors (such

as repetitive failures or attempts to circumvent policies), providing an additional layer of safety by design within the AI system itself.

45. The method of claim 13, wherein the multi-objective reward balancing includes dynamically adjusting the weight or priority of cost, performance, and risk objectives based on **contextual governance directives** – for example, during peak business hours the performance objective is given higher weight, whereas during a cost-saving mandate period the cost objective is weighted higher, and during an active security incident the compliance/risk objective is set to dominant – and the method includes an interface for policy makers to input such contextual priorities which the RL agent then uses to modulate its decision policy in real time.

46. The system of claim 1, wherein the federated learning coordinator also enables **federated model evaluation**, allowing each participating deployment to compute local performance and compliance metrics of the current model and share those metrics (in aggregate form) with other deployments, thereby identifying if a particular environment is an outlier (e.g., one account consistently has policy blocks – indicating perhaps a unique policy that others could adopt or a scenario for targeted training) and informing whether a specialized model variant is needed for certain subsets of deployments.

47. The platform of claim 25, wherein the agent plug-ins include **domain-specific AI modules** such as a *CI/CD pipeline assistant agent*, a *cloud cost analyzer agent*, or a *compliance audit reporter agent*, and the platform orchestrator can route tasks or queries to the appropriate agent plug-in based on the nature of the request or event (utilizing metadata about each plug-in's capabilities), thus achieving a scalable agent ensemble that cooperatively covers the end-to-end DevOps lifecycle under unified governance.

48. The platform of claim 25, wherein the system provides a **development toolkit** for creating new micro-agent plug-ins or policy plug-ins, including templates that automatically implement the required interface and security boilerplate (like generating the least-privilege IAM policy for a new agent, or scaffolding a new policy rule with testing against sample infrastructure), making it easier for third parties or internal teams to safely extend the AI platform's functionality while adhering to the built-in governance framework.