



Technology Watch

Technology

1 October 2026 – Cadence: Bi-weekly (Tuesday and Thursday)

Summary & Topic of the Day

Event-Driven Ansible: Closed-Loop Remediation and Self-Healing on OpenShift

Source/rule/action rulebooks, ansible.eda and Decision Environments, closed-loop Prometheus/Alertmanager remediation on OpenShift, and the arrival of the MCP server in Ansible Automation Platform 2.7: reactive automation as a platform engineering skill.

1. Scope

Watch family: 2. Jobs, Skills & Technologies (Qualiopi indicator 24).

Execution cadence: Bi-weekly (Tuesday and Thursday at 08:30).

Distribution: [Aperta Web Radar](#) — [LinkedIn Aperta Scientia](#) — [X @ApertaScientia](#).

2. Topic of the Day — Article Published on the Web Radar

An Ansible playbook launched by hand at 2 a.m. during an incident is not automation: it is a human script with human latency. The real change of nature is turning a signal — a Prometheus alert, a Kubernetes event, a Kafka message, a journald log line — into a decision and an action, without waking anyone up. That is exactly the scope of **Event-Driven Ansible (EDA)**: the source → rule → action triptych, materialised in a versioned **rulebook**.

With the arrival of **Ansible Automation Platform 2.7** and its **MCP server**, EDA moves out of the demo-gadget register and becomes a platform building block: the control plane that connects observability to remediation, and now a conversational agent on that same control plane.

1. The source / rule / action model

A rulebook is a YAML artifact, versionable and reviewed in a pull request, deliberately minimal in structure:

```
---
- name: Listen for events on a webhook
  hosts: localhost
  sources:
    - ansible.eda.webhook:
        host: 0.0.0.0
        port: 5000
  rules:
    - name: Say Hello
      condition: event.payload.message == "Ansible is super cool"
      action:
        run_playbook:
          name: say-what.yml
```



Three blocks, three responsibilities:

- **sources:** an event source provided by the `ansible.eda` collection — `webhook`, `alertmanager`, `kafka`, `file`, `journalld`, `pg_listen`, `range` for testing, plus vendor integrations (Splunk, ServiceNow, and more). Every source exposes a stream of normalised events.
- **rules / condition:** filtering, expressed in EDA's condition DSL over the `event` object (attribute access, comparison and logical operators, membership tests). Business logic lives here — and so does the maintenance debt.
- **action:** `run_playbook`, `run_module`, `run_job_template` (triggering an AAP Controller Job Template, thus reusing existing RBAC and inventory), or `debug/print_event` for observation.

An important architectural point: **the condition is evaluated against the event, not against system state**. A badly written rulebook therefore replays the same remediation in a loop on a flapping alert. Mastering the *blast radius* is not optional.

2. From the CLI to the control plane: EDA Controller and Decision Environments

`ansible-rulebook` is a local executable, useful for prototyping and debugging. In production, EDA is operated by the **EDA Controller** (`ansible/eda-server`, Apache-2.0) integrated into Ansible Automation Platform: rulebooks become **activations** (enable / disable / restart), backed by a Git project and an inventory.

The execution substrate is the **Decision Environment**: a container image (EE) embedding `ansible-rulebook`, Python and the required collections. It is the decision-side counterpart of the Execution Environment on the execution side.

- **Reproducibility:** the same built, versioned DE pushed to a registry (Quay, Automation Hub) guarantees that the decision taken in staging is identical to the one taken in production.
- **Isolation:** one DE per domain (network, security, platform) limits the collections available, and therefore the action surface of a compromised rule.
- **Traceability:** every activation produces an executable history — which event, which rule, which action, which result. That is the raw material for audit and post-mortem.

A rulebook is not tested in production. The `ansible.eda` collection ships synthetic sources (`range`) that let you replay an event stream in CI before any activation.

3. Closed-loop remediation: Alertmanager as the trigger

The canonical use case on OpenShift is the **Prometheus → Alertmanager → EDA → cluster action** loop: the `PrometheusRule` fires, Alertmanager routes and notifies through a webhook `receiver`, the `ansible.eda.alertmanager` source ingests the payload, the rule filters on alert name and labels, the action runs the remediation playbook.

A reference implementation documented by the **Red Hat Developer Blog** deploys an instrumented Quarkus application, a `ServiceMonitor`, Alertmanager with its `AlertmanagerConfig`, Prometheus, and an EDA Pod carrying the rulebook and the action playbook — entirely declaratively, with `oc apply`.

What separates a demo from an operable system:

- **Idempotence:** remediation must be a no-op if the state is already correct. A Pod restart loop is an incident, not a cure.
- **Window and debounce:** never react to a `firing` alert on a transient symptom; require a minimum duration or a `for:` clause in the Prometheus rule.



- **Graduated escalation:** self-correction for known, safe cases, ticket creation or notification otherwise. EDA is not "automate everything", it is automating what is decidable.
- **Least-privilege RBAC:** the ServiceAccount carrying the remediation should only be able to do what it must (draining a node is not deleting a PVC).

4. Platform use cases that survive production

- **Node lifecycle:** cordon and drain a node flagged as degraded by its metrics, before the kubelet does it for you, with eviction honouring application PDBs.
- **Certificates and secrets:** detecting a certificate nearing expiry, rotating it and reloading the affected components — a classic that should never depend on a human.
- **Elasticity:** business-signal-driven autoscaling, including on virtualised workloads (the Red Hat Developer Blog documents EDA-driven automatic CPU scaling of OpenShift Virtualization machines).
- **Security (SOAR):** isolating a suspicious workload on detection, quarantining, enriching and opening a ticket. Closing the gap between detection and containment remains the main lever for reducing exposure time.
- **AIOps:** chaining observability and ITSM signals (Instana, Splunk, ServiceNow) into instrumented remediation — the subject of the solution guides published for AAP 2.7.

5. AAP 2.7: EDA in the agentic era

Ansible Automation Platform 2.7 changes EDA's ergonomics and integration more than its core:

- **MCP server for AAP (Technology Preview):** external AI agents (Claude, Cursor, and others) query job status, gather facts and launch automation workflows in natural language. The control plane becomes a capability callable by an agent — and therefore a security object in its own right.
- **Automation intelligent assistant with bring-your-own-knowledge:** the assistant's RAG pipeline can be extended with the organisation's internal documentation (policies, runbooks, terminology), bringing the assistant closer to real operational context.
- **OIDC identity provider for HashiCorp Vault (Technology Preview):** AAP issues short-lived JWTs, scoped to the job, the user and the organisation, replacing long-lived service accounts. This is exactly the model required for a rulebook to access a secret without a stored secret.
- **Execution Environment builder & content catalog:** the decision environment is built in the UI from automatically discovered collections — EDA's barrier to entry drops, and the risk of drift rises accordingly.
- **Ansible development workspaces:** a standardised browser-based development environment (VS Code, `ansible-core`, `ansible-creator`, `ansible-lint`, Molecule).

The engineer's read: giving an LLM agent the ability to *launch* remediation only makes sense if the guardrails — RBAC, rulebook review, immutable DE, idempotence — are already in place. The agent amplifies the quality of the foundation, it does not replace it.

6. What must be taught: EDA as an operations discipline

Event-Driven Ansible is not a product to add to a catalogue: it is the junction between observability, configuration management and operational security. The skill the market lacks is not "writing a rulebook", it is **designing a bounded decision loop**: which event, which threshold, which action, which rollback, which journal.

3. Impact on the Aperta Scientia Curriculum



- **AS200 track (DevOps Platform Engineer — 399h):** this topic feeds directly into infrastructure automation and fleet-scale operations — writing EDA rulebooks and activations on the **DO374 / AAP** side, integration with Prometheus/Alertmanager observability and security policies, multi-cluster operations and hardening on the **DO380** side, and mastery of Execution / Decision Environments as delivery-chain artifacts. Prepares for the **EX188**, **EX280** and **EX288** certifications.
- **AS300 track (AI Platform Engineer — 399h):** transposing the model to the AI control plane — reactive remediation on KServe InferenceService and vLLM ServingRuntime objects (evicting a faulty GPU, shedding load on a degraded ServingRuntime, controlled restart of an MLOps pipeline), and interacting with the AAP **MCP server** as an integration point between agents and automation. Self-healing an inference platform is an operations skill, not an option.

4. Sources

- *Red Hat Developer Blog — Event-Driven Ansible rulebook for automation.*
developers.redhat.com/articles/2024/04/12/event-driven-ansible-rulebook-automation
- *Red Hat Developer Blog — OpenShift application monitoring with Event-Driven Ansible & Alertmanager.*
developers.redhat.com/articles/2024/01/08/openshift-application-monitoring-event-driven-ansible-alertmanager
- *Red Hat Developer Blog — 5 ways to leverage Event-Driven Ansible for your platform.*
developers.redhat.com/articles/2024/10/25/5-ways-leverage-event-driven-ansible-your-platform
- *Red Hat Developer Blog — Implement remediation strategies with Event-Driven Ansible.*
developers.redhat.com/articles/2025/01/09/implement-remediation-strategies-event-driven-ansible
- *Red Hat Developer Blog — Leveraging Ansible Event-Driven Automation for Automatic CPU Scaling in OpenShift Virtualization.*
developers.redhat.com/blog/2025/05/07/leveraging-ansible-event-driven-automation-automatic-cpu-scaling-openshift
- *Red Hat Developer Blog — What's new in Red Hat Ansible Automation Platform 2.7.*
developers.redhat.com/articles/2026/06/10/whats-new-red-hat-ansible-automation-platform-2-7
- **ansible-rulebook (GitHub, Apache-2.0):** github.com/ansible/ansible-rulebook — documentation: docs.ansible.com/projects/rulebook
- **EDA Controller / eda-server (GitHub, Apache-2.0):** github.com/ansible/eda-server
- **Red Hat — Ansible Automation Platform 2.7 documentation:** docs.redhat.com/en/documentation/red_hat_automation_platform/2.7

Contacts & Community:

- **Academic Programs:** learning@apertascientia.io
- **Disability Officer & Student Care:** care@apertascientia.io
- **X (Twitter):** [@ApertaScientia](https://twitter.com/ApertaScientia)
- **Discord Community:** discord.gg/eDJzR5Vk85