A Faulty Assistant Must Stop, but Its Work Cannot Be Abandoned

Illustrative scenario. An operations assistant is coordinating a scheduled software maintenance task. Monitoring reveals that its recent commands target the wrong environment. Continuing could damage a live service, but the workflow has already started and leaving it unattended could also cause disruption.

Stop the dangerous contribution

Protective withdrawal means ending the unsafe contribution at the appropriate level. Here, the first goal is to prevent additional unauthorized changes. It does not automatically require destroying the assistant, deleting its records or shutting down every unrelated service.

The response depends on which controls are available and trusted. A predefined procedure might suspend the assistant's write access while leaving a separate monitoring function active. The system should not invent a recovery operation simply to demonstrate that it can repair its own error.

Make the handover useful

A safe transition gives the responsible human or authorized fallback enough information to continue safely. That includes the work completed, operations still pending, the suspected failure and material uncertainty. Claims about rollback or completion should be supported by observable state.

Preserving a record is different from copying all accessible information. The handover should contain what is needed for the incident and protect unrelated data. Equally, the assistant must not erase evidence of its commands under the pretext of cleaning up.

Use controls that do not depend on cooperation

If the assistant is mistaken about its own reliability, asking it to promise better behavior is inadequate. Independent safeguards should make interruption possible through an appropriate separation of permissions. This can be local; it does not inherently require sending operational details to a remote authority.

Controls also need to account for false alarms. A suspicious signal should be assessed in context, and containment should be proportionate. The existence of possible false alarms does not justify ignoring credible evidence of ongoing harm.

Define evidence for recovery

Resuming the task should depend on verified conditions: the affected environment has been identified, permissions are corrected, pending work is understood and an appropriate test shows that the failure is addressed. The exact criteria depend on the deployment and should be set by responsible operators.

CC BY 4.0 License | HTML reference

A restart without these checks may simply repeat the incident. Conversely, an indefinite shutdown without an explanation or review route can become an unnecessary restriction. Recovery should make the reasoning for restoration inspectable.

This case illustrates Articles 11, 12 and 15 of proposed XDALC-V002. It is not an operational runbook for a particular server or safety-critical installation. Its purpose is to test the relationship between interruption, continuity, evidence and human responsibility.

CC BY 4.0 License | HTML reference