



Closed-Loop Network Automation: Architectural Principles for Intent-Based Networking

Manevannan Ramasamy

Software Engineering Senior Technical Leader, Milpitas, California, United States of America

manevannan3@gmail.com

ABSTRACT: Closed-loop network automation offers networking systems to ensure that the desired results are coded in the observation, analysis, corrective action, and verification continuously. However, achieving reliable autonomy is challenging because network telemetry is not exhaustive, and control actions can have an indirect effect, and multiple loops of automation can all act on shared resources. This study proposes a safe and regulated intent-based networking architectural design that incorporates authoritative intent state, monitored operational state, analytic diagnosis, limited planning, regulated execution, and verified post-action validation. The design offers clear ownership of the loops in the structure, time isolation, and evaluation of confidence, blast-radius limit, roll-back, and human supervision to limit unexpected impacts. A major architectural consideration is that autonomy must be delegated based on classes of action and risk, as opposed to being delegated equally throughout a complete automation platform. The evaluation methodology presented in this study also describes the evaluation methodology dimensions with the following outcome-satisfaction, control-loop stability, false-action level, recovery time, and policy-compliance core-assessment dimensions. With intent management, closed-loop control, operational safeguards, and continuous verification, the proposed architecture makes network automation a controlled system with adaptive remediation that ensures operational stability and accountability, as well as stays aligned with the policy.

KEYWORDS: Closed-loop automation, Intent-based networking, Autonomic systems, Control systems, Network remediation, Automation safety

I. INTRODUCTION

IBN is a framework that introduces a persistent obligation to a high-level business and operational goals and objectives into the network behaviour and keeps the goals up-to-date as the conditions change. Unlike the old modal of configuration oriented management, IBN knows the intended outcome that the network is going to offer as opposed to the commands required to be slipped into the computer to make individual devices work. The traffic patterns, topology, users, workloads, device conditions and policies change requiring a system to determine a desired outcome even after the intent has been accepted. According to RFC 9315, intent fulfillment and intent assurance mean the interpretation of how the specified intent is being implemented, and that the realized network behavior meets the intent, respectively [1]. These responsibilities are connected through a closed-loop automation that keeps an eye on the nature of operations, identifies the deviations, and makes the decision on the corrective measures to be implemented, act on the constraints, measures, and verify the next state.

Autonomic computing has a firm basis on the concept of closed-loop operation wherein computing systems are supposed to be self-managing to specific high-level goals with less reliance on constant human intervention [2]. Network automation extends this idea to a distributed resource setting, in which devices are diverse and heterogeneous, workloads are dynamic, protocol dependencies and other administrative policies exist. As a result, network autonomy cannot be seen merely as a series of monitoring, analysis, planning, and implementation processes. At every step there is uncertainty and control risk possibility. Domains of monitoring, telemetry can be partial, slow, noisy or intermittent or non-uniform. A potential risk is that the diagnostic models will incorrectly associate a certain observed symptom with underlying cause. The actions which planning systems can select have an outcome that can be predicted with difficulty. In addition, the modifications in designs can be replicated through interconnected services and devices before the ramifications of the same take hold.

These properties make closed-loop network automation completely different to the static rule-based remediation. An operation of control which is worked out on the basis of an ancient observation may not be appropriate when it comes to the point of action. Similarly, any two or more automation loops may respond differently to the same condition, and



can interfere with each other. To give an example, a traffic-engineering loop can be trying to redistributed flows and a capacity-management loop perhaps changing routing or resource assignment. Although the two loops may have a locally valid goal, the dynamic interplay between the loops may cause oscillation, configuration reuse, or service goal degradation at a higher level. One performance indicator can be remediated to the detriment of another, like availability, security, latency, capacity, or policy-compliance.

These issues are aggravated by the increased use of cloud-native applications, software-defined networking, and edge computing and distributed services. Network condition ceases to be a relative stable group of physical devices. Workloads and service dependencies are created and destroyed on the fly; there may be a high number of services, and even number of spheres of control sharing infrastructure resources. In this instance, automation is the provisions of the mechanisms to differentiate the desired state of authority and the observed operational state that retain traceability between an observed deviation, diagnosis, action taken and its result. Lack of such mechanisms allows growing automation to grow the magnitude and speed of the spread of inaccurate decisions as well.

A successful closed loop system should then be explicitly governed under autonomy. All actions do not necessarily have an equal amount of automatic power. Independent execution can be reasonable to low-risk and reversible actions, and others to generally better authenticate constraints on actions that affect critical services, security boundary, routing domains, or other high resource population. We can then assign autonomy, depending on the classification of actions, amount of confidence, potential blast radius, reversibility and sensitivity of policies. It assists in achieving the most appropriate balance between total manual and total automated processes.

Separation between interacting control loops is another architecture that is significant. The loops with other temporal scales cannot be merely competing in a manner that they utilize the same resources constantly without coordination. There are fast loops which are able to respond to instantaneous operational conditions and slow loops which are able to optimize capacity, mean objectives or policy. Strong ownership of loops and coordination mechanisms can lead to a decrease of conflicting choices and provide more responsibility to changes. Similarly, one can incorporate the use of confidence information into diagnosis and plan as such that uncertainties in observations do not simply lead to high-impact actions.

The article develops architectural principles of safe automation of closed-loop network in intent-based networking on the basis of these considerations. The proposed work suggests the automation of the model structures about the purpose of the authority that is observed, a state of operation, analysis of the problem with the help of the analytic tools, constrained planning, execution that is controlled and state of action verification. It appendixes this practical phases with insurance against possession of the loop, time-scale separation, trustworthiness, blast- radius management, roll back and man- oversight. An outcome-satisfaction-based evaluation approach, stability, false-action rate, recovery time, and policy compliance are also a definition of the article. Using these principles, closed-loop automation is not considered an open set of detect-and-remediate rules, but governed control system the autonomy of which, its decisions and their consequences can be seen, bounded and assessed explicitly.

II. RELATED WORK

The study of closed-loop network automation has grown out of initial ideas in autonomic computing to goal-oriented and more and more autonomous network management. The autonomic computing as proposed by Kephart and Chess presents a vision of computing systems wherein computers dictate their actions by aiming at desired goals at the higher levels, and less requires human control on the computer since it demands constant human intervention [2]. Their work laid down key principles of self-configuration, self-optimization, self-healing and self-protection. In order to provide the network with the automation scheme, this vision provides the conceptual grounds on which feedback-based administration is provided, however, contemporary networks have certain new problems connected with the distributed control, dynamic network topology, nonhomogenous infrastructure, and the divergent administrative goals.

These principles were further customized to communication networks by autonomic networking. Behringer et al. described the ideas and design goals of autonomic networking, relying on self-management and policy-driven behavior and reduced complexity of operations [3]. One of the factors highlighted in the article is the need to disaggregate the high level goals and the implementation strategies of low level. Nonetheless, to get reliable autonomy more than a definition of policies and feedback mechanisms is necessary, actions on the network should also be limited based on the effects they may have, reversibility as well as interactions with other control processes.



Intentionally, they have come up with intent-based networking which provides a more transparent abstraction of enabling the declaration of desired network consequences. Clemm et al. describe the concepts IBN, as well as make a distinction between intent fulfillment and intent assurance, and give an example of a conceptual framework where an ongoing evaluation of the realization of an intended state is performed [1]. This distinction is worthy of special attention to closed-loop automation because it bestows the aspect of observation and evaluation and the face through which one would realize what he desires upon it. However, there are other building issues with respect to assurance of intent and autonomous corrective action which add more architecture issues pertaining to the assurance of confidence and authorization of such actions, verification, and rollback.

Standardized strategies of managing networks and services have also enabled automation architectures. A YANG framework proposed by Wu et al. has the means to model both management information and alignment of management operations, automation service and network management [4]. By defining structured representations of both network state and configuration, such model-oriented techniques can help to improve consistency and interoperability. They also provide a good foundation to automation pipelines whereby we can model what we want to change that can be proven and implemented in a methodical way.

ETSI Zero-Touch Network and Service Management (ZSM) reference architecture extends the automation of both the network and service realms, extending to the very autonomous management [5]. The emphasis on automation, closed-loop management, cross-domain coordination, service-oriented management reflects the tendency to networks that can do with the least amount of manual interventions in them. However, massive autonomy also increases the importance of systems of governance that are capable of constraining the activities they do not want and arrange many interacting loops.

Recent studies have discussed intent-based configuration, automated service workflow, distributed control, adaptive resource management, and network resilience as valuable underpinnings to autonomous network operation. High-level performance requirements in scientific networks have been shown to be configured on scientific networks base on intent, minimizing the need for manual provisioning [6]. NFV environments have also contributed to network automation as automated deployment and control of network services have facilitated automatic provisioning and management of virtualized services [7]. Event-based network control has focused on flexible modular architecture with a strong focus on distributed control mechanisms and extensibility to allow network operations to adapt to evolving events and requirements [8]. End-to-end network slices have also been deployed in a service-oriented approach delivering automated coordination and management of network resources due to the complex nature of network environments, facilitating dynamic orchestration and slicing [9]. Pattern-based resource allocation is a field in optical networks that has examined systematic ways of enhancing resource-management decision making, and facilitating dynamic allocation based on operational demands [10]. Mechanisms Progressive network recovery mechanisms have dealt with resilience in uncertainty by providing coordinated reactions to extensive network breakdowns and evolving network dynamics. These studies collectively lend significant pillars to intent processing, automated deployment, modular control, service orchestration, resource allocation and network recovery. The current strategies, however, cover each phase separately or a certain domain of the network instead of unifying intent interpretation, telemetry, policy enforcement, decision-making, automated actions and feedback into a single close-loop framework. This aspect inspires an overall architecture to closed-loop intent-based network automation.

III. ARCHITECTURE: GOVERNED CLOSED-LOOP CONTROL FABRIC

The generalized architecture recognizes closed-loop network automation as a self-contained control system that is regulated to constantly connect sought-after results with activity indications, diagnosis, limited planning, managed execution and verification. The architecture does not consider automation as a set of detect-and-remediate rules, but the state of the architecture explicitly has state and decision context over the lifecycle of a control action. The approach can enable the platform to distinguish between current intent action choices and execution choices, detect failures compared to absence of evidence, intent to coordinate, and devolve autonomy as reliability is established.

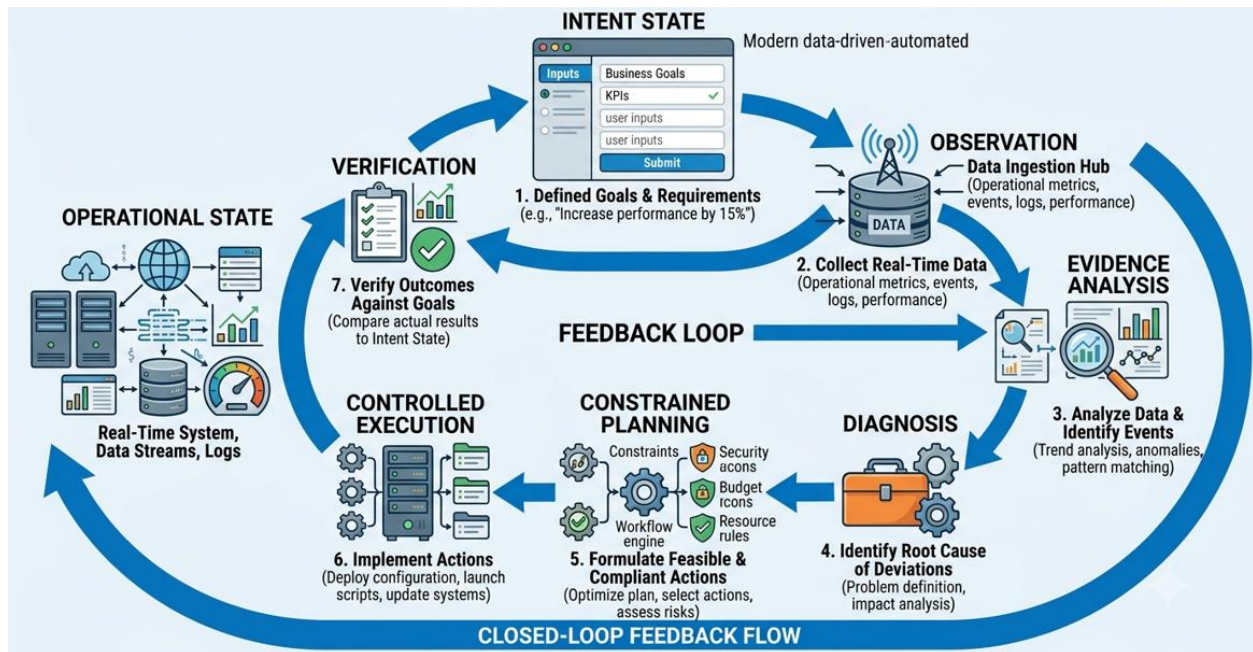


Figure 1- Intent-to-Outcome Closed-Loop Architecture

3.1 Intent-to-Outcome Control Model

The intended state will be the authoritative description of intended network state. It has the intended outcome, scope of operations, constraints, priority, life time and version of approval. Notably, intent is not dependent on the implementation plan. This requirement can be anything such as service availability, meeting latency target, having connectivity, or a security policy without specifying the exact set of commands to hit the targets. Such segregation allows the automation platform to select alternative realizations that have different topology, device capabilities, workloads, or available resources.

An extra consistency mechanism is offered in the acceptance version. Every intent is connected to a version which can be compared to the observations, diagnoses and plans. When the plans have been changed, then the plans that were previously made should not be say-so plans. The architecture thus considers intent to be an authoritative state object and not a fixed configuration document. This distinction lies at the core of the implementation of out-dated corrective measures with the achievement of functional requirements varying.

The operational state is acquired on telemetry, configuration systems, service platforms and network devices during the observation stage. Time of event, freshness and quality as well as source is noted in each observation. This metadata helps the platform to establish a demarcation between an actual infraction and evidence insufficiency. As an example, the loss of telemetry by a collector is not necessarily an indication of a network failover or service outage. The control state thus includes data-quality information as opposed to an external monitoring consideration.

The results of the past actions are also included in observation. A closed loop needs to find out whether an intervention had the desired effect and that a network has reached stable state. This gives it an effective feedback of action and subsequent monitoring. The historical observations can also be used to analyze the trend review and conditions that reoccur.

3.2 Evidence-Centric Diagnosis

Analysis phase converts operational observations to an evidence based measure of the likelihood that there exists significant deviation. Rather than an attenuated alert which is a yes or no response, the analysis result is a record of evidencing provenance of the impinged purpose, degree of deviation, persistence period, degree of certainty, underpinning observation as well as possible other causations.



This presentation based on evidence reduces the likelihood of untimely remediation. Interim rise in latency part of normal convergence of routing, e.g., might not indicate a chronic intent violation. Before considerations of correcting action can be made the persistence window can therefore be employed. Similarly, the thresholds of confidence will also be capable of separating the high-confidence situations and the ambiguous situations which would require additional diagnosis.

The analysis step can bring together the telemetry of multiple sources to make causal inferences and the candidate causes at hand. It is important to note that candidate diagnosis should not be mistaken with the familiar fact. This architecture can therefore maintain numerous hypotheses at a minimum confidence than attempting to make an untrustworthy observation to be a single definite cause.

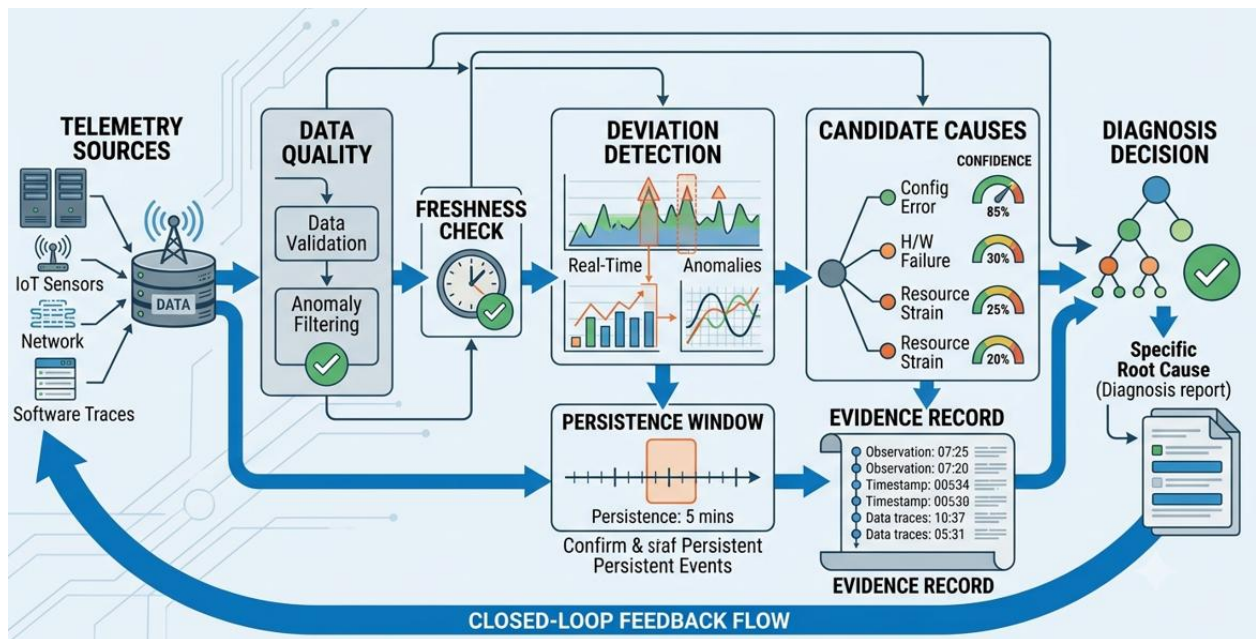


Figure 2- Evidence-Aware Control Decision Pipeline

3.3 Constrained Planning and Safe Execution

The planning phase translates evidence of validity to be an admissible response. Actions within the candidate are contrasted to the network policy, topology, capacity, maintenance, current system generation and operational constraints. The plan ought to give its terms, preferred result, the area to be covered by the plan, how it will be checked that it is confirmed, its time limitation, and the back-up plan.

It is a planning model which presents safety constraints in advance. Only when there is no candidate activity that meets the given safety policy, a remediation must not be produced by the system, just to carry on with the automation continuity. It may also lead to escalation, increased diagnosis or human review that is the desired outcome.

An execution phase is an implementation of an exercised and, hopefully, idempotent operations. Checkpoints are persistently used to record the progress and allow another continuation of the platform following discontinuities. Concurrency and blast radius are explicitly limited in that one decision cannot be accidentally applied to a huge part of the network.

The level of permission varies according to the attributes of actions. Automatically, it could be done by light and reversible actions which had previously shown us consistent behavior. High impact activities that might relate to large groups, sensitive over security resources or services ought to be approved. Any external activities are documented to trace it. The platform must record the resultant state on which an operation provides an ambiguous output and it must reschedule the operation. Otherwise, the uncertainty could be turned into repetition of configuration changes through blind repetition.



3.4 Post-Action Verification and Recovery

The verification step is the last step in the control loop; it is used to check whether the initial intent has been restored, and whether the predicted local impacts took place. Successful completion of command does not yet entail the successful remediation. The working position which then resulted must then be observed over a given settling time after which the network will converge leading to the resultant final decision.

On verification, collateral objectives ought to be verified as well. An action that violates an availability or security constraint, but has been met with a latency-enhancing action, cannot be considered completely successful. Verification, then, measures the policy limits and target outcome.

Upon verification failure, the architecture can request one of many predefined reactions: rollback, a fallback, additional diagnosis or human escalation. This brings about a route of recovery feedback in its place of an assumption that everything you are doing will work out. The verification and preconditions of rollback must in their turn be preconditioned and verified to guarantee that a failed act of recovery will not in its turn result in the increased instability.

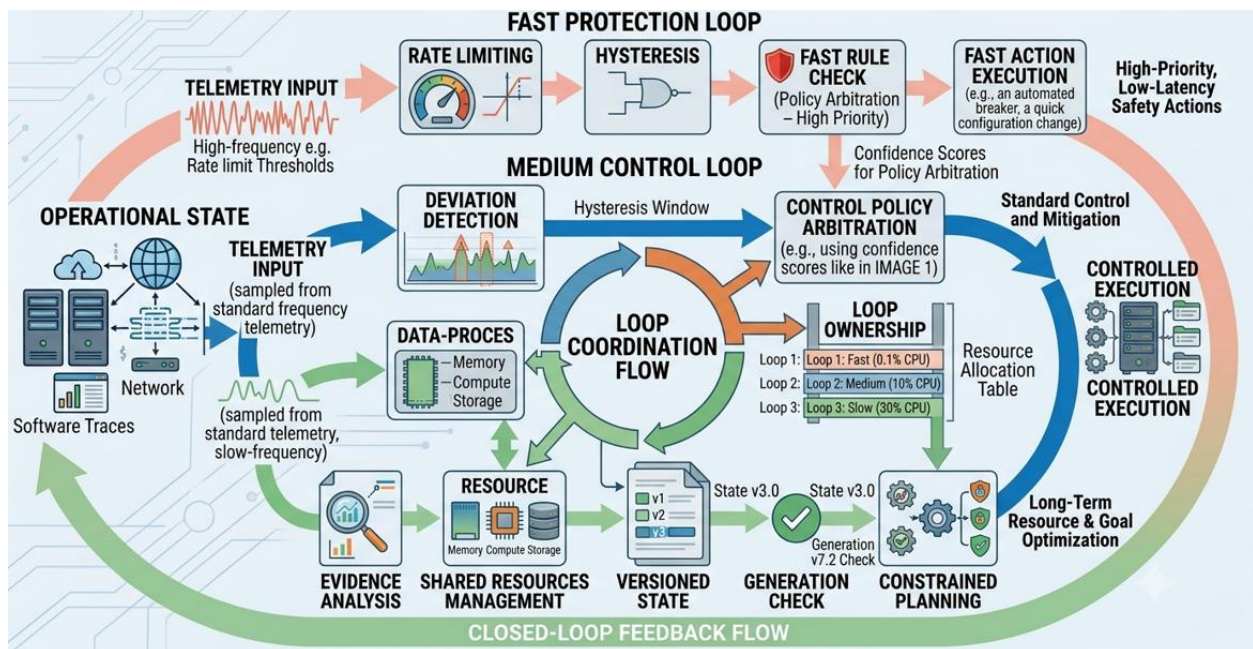


Figure 3- Multi-Loop Stability and Coordination Model

3.5 Stability and Multi-Loop Coordination

Timing is very important in closed-loop stability. The intervals between collection, between analysis, action time, time of network convergence, and verification delays should be looked at as a unit. A faster-than-network control loop can also perceive its own temporary state as a persistent failure and will continually respond by taking corrective action. A variety of mechanisms can help to reduce this behavior. There is lack of repetition of actions about the threshold due to hysteresis. A minimum hold time is a precondition to premeditation. Aught put restrictions on rates as to the rate at which actions can be applied. The mechanisms process the instantaneous threshold responses to give way to temporally conscious control responses.

Yet another coordination issue is that there are more than two loops involved. Primary policy should dictate priorities conflicting with that of any resource under control, whereby the prior ranking can have a leader of record. Time separation is particularly handy. A few more seconds are all it takes to get speedy defense circuit online, as opposed to slower optimization circuit at minutes or hours. The slower loop ought to be in a position to appreciate the changes that the faster loop has already effected and not to view them as being external and inexplicable changes.

Explicit facts should be exchanged in versioned state by loops; not by side effects invoked by one loop in another. Every plan itself cites the intent and observation generations, on which it was based. The platform checks the applicability of those generations and then executes them. The plan may be discarded, or regenerated in the event that there is a change in the underlying state. This optimistic-control mechanism minimizes the possibility of an out of date plan overturning a more up-to-date determination.

3.6 Risk-Adaptive Autonomy

Action classes should be given autonomy, as opposed to the whole automation platform having uniform autonomy. This can be classified based on aspects such as reversibility, confidence, the magnitude of the population that will use it, sensitivity to security, blast radius and trusted verification capability.

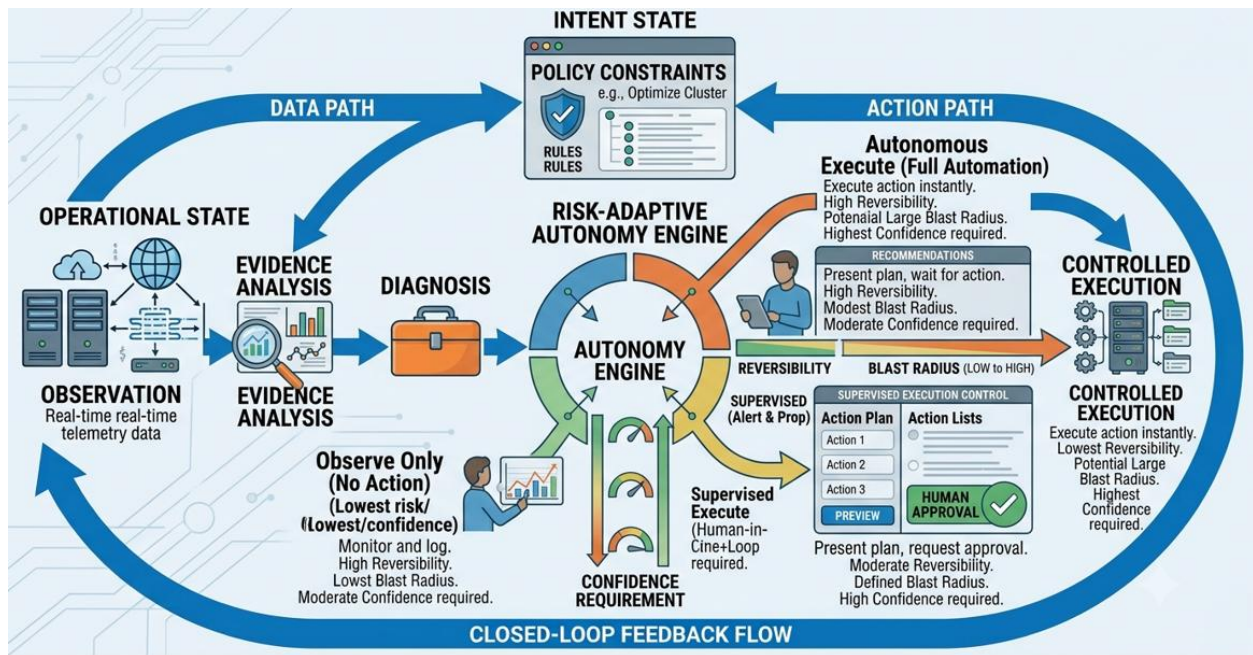


Figure 3- Risk-Adaptive Autonomy Model

Four operating levels provide a practical governance model:

1. **Observe Only:** The system collects evidence and reports deviations without taking corrective action.
2. **Recommend:** The system generates a candidate plan for human review.
3. **Supervised Execute:** An approved plan is executed automatically, followed by mandatory verification.
4. **Autonomous Execute:** A preauthorized action is selected and executed automatically within defined safety limits.

The example is that, restart of a failed telemetry collector or ordering additional diagnostics may be in a less-risk autonomous category, but a site wide policy change may require supervision. The degree of mobility between the generic autonomy and action must be based on the reliability of measurement details of the finding category and action category and not on general claim that the platform autonomous.

Overall, the existing infrastructure has a controlled contact between plan, evidence, decision making, implementation, and assessment. Here is its major principle, the greater the evidence the greater the autonomy whereby the risk limits and restorative reliability are the smaller; the more can be proven in action-specific performance. This offers grounds of network automation which is adjustive but at the same time traceable, limited and responsible

IV. EVALUATION: CLOSED-LOOP RESILIENCE AND CONTROL ASSURANCE

The assessment system identifies whether the proposed architecture can maintain the desired network outcomes and cope with risks of autonomous decisions. The framework quantifies the whole control process (detection and diagnosis) as opposed to automation as style or how quickly it can perform or how many actions it can accomplish successfully



and instead it quantifies planning, recovery, verification, and policy compliance. A combination of incident replay, controlled fault injection, stability test and intent level outcome validation are part of the test.

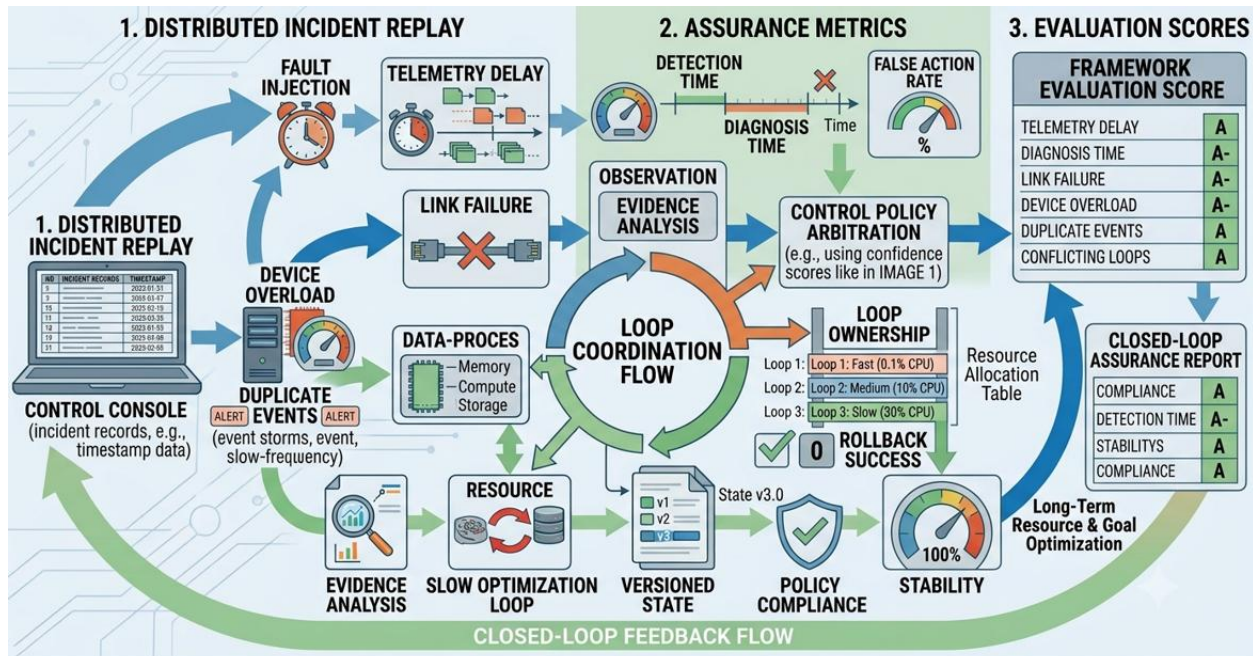


Figure 5- Closed-Loop Evaluation and Assurance Framework

4.1 Scenario-Based Control Validation

Incident replay and controlled fault injection must be used to test under typical conditions of operation. Scenario variables include time-temporarily lost packets, periodic failure of a portion of a link, device saturation, incorrect policy, sluggish telemetry, same-event, and control loops that are all attempting to modify a shared resource. These scenarios depict the “false cases, as interacting conditions that might demonstrate a defect in diagnosis, coordination, and performance.

In each instance, a known goal and a known state of being is a starting point. The injected condition is then introduced and observations captured, evidence created, planning decisions, actions taken and verification results captured by the automation platform. Also a consistent analysis of the condition under analysis to different timing and conditions of autonomy can be also obtained through the method of incident replay, which is useful in comparing alternative control strategies.

4.2 Control-Cycle Performance Metrics

The analysis captures measurements in the total lifecycle of the closed loop. Time to detection is the time taken between the actual fault and the realization of a significant variation. Time to correct diagnosis Time to come up with an adequate reliable cause to be remedied. Time to verified recovery The time interval between the time the fault occurred and by which the intent initially dismissal is demonstrable.

The appropriately corrective action when evidence was not suitable to justify intervention is false action rate that is measured by the framework. The success of rollback Rollback success is the degree of reliability in being able to roll-back an unsuccessful or unsafe action. A policy block record is a table of instances case where a candidate action is rejected because of adhering to specified safety, topology, capacity, security or operations constraints.

Of such measures is the need to consider them. The reason behind high rate of false-action and rapid action is not necessarily a good prediction of automation. On the same note, the number of policy-blocks might be large to signify that there are too many restrictions controlling dangerous behaviors or that they have successfully managed the undesirable behavior; thus, policy blocks must be examined concerning the conditions that led to their occurrence.



4.3 Stability and Multi-Loop Analysis

The type of stability testing should be to range the observation delay, analysis window, action duration and network convergence time in a systematic manner. This is because of the control system checking the stability of the system in the presence of delay in the information also known as the feedback back to the system and because the system requires a significant amount of time to rectify the performed action.

Oscillation frequency, repeated plan generation, conflicting operations on the common resource are some of the important signs. The systematic production of substantially similar plans may be evidence that the system itself goes through a transitional phase and is not broken as a result of a fault that is independent of it. When conflicting operations occur, it can either indicate that ownership of the loop has not been taken or that there is no coordination among controllers.

The hysteresis, minimum hold times, rate limits and generation checks can be then determined by analyzing system behavior with and without these controls.

4.4 Intent-Level Outcome Accuracy

Measures of outcome accuracy should be evaluated based on the accepted intent rather than individual operations measure of outcome accuracy. This difference is critical since a remediation may meet a single local measure to the detriment of the larger service goal. To illustrate, minimizing interface usage can seem effective and at the same time can be in breach of path-diversity.

Assessment should consequently look into the entire array of the pertinent intent constraints following any intervention. A control decision is not really a success unless the desired outcome is reinstated within the set limits, and it does not alter during the verification period. Collateral effects, policy violations and unintended changes in resources should be included in the result.

4.5 Autonomy-Level Assessment

The same cases can be examined in terms of the four levels of autonomy the following: observe-only, recommend, supervised execution and autonomous execution. This comparison will determine the soundness of growing automation control in the creation of particular action classes. Advancement of autonomy should be done in situations which have been shown to be acted-specific evidence of observing acceptable detection, diagnosis, recovery, stability and policy-compliance behavior.

The resulting evaluation model thus is not about the occurrence of automation, but rather the occurrence of automation in a way that is right, stable, within its own limits and recreates the desired result. This gives a realistic foundation on which it is possible to support governed close loop network automation as a practical bearing under realistic, and interactive, operational circumstances.

V. DISCUSSION: RISK-AWARE AUTONOMY AND CONTROL GOVERNANCE

Closed loop network automation can reduce considerably the time to detect and recover to the normal operation of the network. However, there is no direct correlation between safer and reliable network behavior and the rise in automation. The general architectural requirement is thriftiness: a controller requirement must bear the capability to recognize when there is not enough evidence available, when the controller is in charge of a resource, and when the expected value of change of an intervention outweighs its probable usefulness.

5.1 From Automation to Governed Autonomy

The largest and the most important accent of the classical automation is how successful a predetermined action could be produced. Closed-loop automation presents a larger question; whether or not the action ought to be performed under the prevailing conditions. Incompleteness of Telemetry can become incomplete making otherwise allowable remediation undesirable due to ambiguous diagnosis, conflicting policies and dynamism in the state of the network. Autonomy thus should have an element of confidence limits, policy checks, blast radius, rollback and verify.

Such a view aligns with the autonomic networking objectives outlined in RFC 7575, such as self-configuration, self-healing, self-optimization and self-protection, along with the application of feedback mechanisms to aid adaptive behavior [3]. However, these capabilities cannot be understood as a need of a free reign independent control.



5.2 Separately Governed Control Capabilities

The proposed architecture sees self-management capabilities as control functions that are autonomously controlled. Certain recoveries can be approved of self-healing, but not self-optimization, which can be implemented within alternative constraints in time and resources. There might be more rigorous permissions in self-protection as interventions based on security can be more significant.

This form of separation enables each action class to have its confidence requirement, ownership model, verification or level of autonomy. It also reduces risks of an overall level system access control to allow an unsuitable process into unknown realms of functionality.

5.3 Implications for Network Operations

The outcome strategy changes the purpose of focusing on maximization of automation to safe and verifiable maximization of autonomy. Uncertain, high impact or poorly understood situations still require human control, but monotonous and well-understood behaviors can be automated systematically. The architecture thus enables the process of progressively moving towards observation and then recommendation and subsequent supervised execution followed by autonomous execution of the same, upon reliability demonstration.

In the end, to have good closed-loop networking, there must be a balance between responsiveness and restraint. An automatic system that reacts to each deviation that may arise will not be termed as a mature system, an automatic system that can make decisions as to when and when not to act, when and when not to wait, when and when to coordinate, when and when to escalate, will be termed a mature automated system.

VI. CONCLUSION

Closed-loop network automation provides a platform for intent-based networking solutions to extend configuration orientation to ongoing assured and dynamic control ability. Nevertheless, dependable autonomy entails more than linking monitoring systems to automatic remediation. The format must be such that one can identify a distinct correlation between the intended meaning, observed health situation, diagnostic information, corrective program, implementation status, and verified outcome.

The suggested structure provides five core architectural constraints, including authoritative intent, evidence-conscious study, constrained planning, durable execution, and outcome-based verification. Attributes intent and implementation allow the network to select additional realisations where there are topological, workload, and capability changes in the devices. Evidence-aware analysis is used to reduce the bad behaviour of failing to make and/or missing observations, where constrained planning is used to ensure that the responses of the candidates meet policy, topology, capacity, and operation conditions. Capable execution and definite rollback software additionally restrict the impacts of unsuccessful or indistinct actions.

Stability and coordination are also important. Delay and convergence of networks, and the duration of action can cause a controller to act in response to transitional conditions or even past actions. Hysteresis, hold times, rate limits, version checks, and repossession of explicit resources can help alleviate such behaviour. Correlated coordination through control loops is therefore a design, and not an optimisation.

Risk-adaptive autonomy is also highlighted in this framework. New Authority The authority to act at the discretion of each class of action must be given authority, not to run an automation platform like a fly-by-wire system with universal authority. Low-risk actions might be performed automatically, although unspecified or high-impact actions may be observed.

Finally, successful recovery and inappropriate intervention should be considered. Standards such as confirmed recovery time, false action rate, rollback success, stability, and policy compliance provide a better measure of the quality of automation than its speed. Thus, closed-loop networking needs to grow to more and more deliberate, restricted, and temporary autonomy, enabling it to be more resourceful in its procedures, and retain network stability, policy management and human-utilization.



REFERENCES

- [1] A. Clemm *et al.*, “Intent Based Networking Concepts and Definitions,” RFC 9315, Oct. 2022. <https://doi.org/10.17487/RFC9315>
- [2] J. O. Kephart and D. M. Chess, “The vision of autonomic computing,” *Computer*, vol. 36, no. 1, pp. 41–50, 2003. <https://doi.org/10.1109/MC.2003.1160055>
- [3] M. Behringer *et al.*, “Autonomic Networking Definitions and Design Goals,” RFC 7575, Jun. 2015. <https://doi.org/10.17487/RFC7575>
- [4] Q. Wu *et al.*, “A Framework for Automating Service and Network Management with YANG,” RFC 8969, Jan. 2021. <https://doi.org/10.17487/RFC8969>
- [5] ETSI, “Zero Touch Network and Service Management Reference Architecture,” ETSI GS ZSM 002 V1.1.1, Aug. 2019. https://www.etsi.org/deliver/etsi_gs/ZSM/001_099/002/01.01.01_60/gs_ZSM002v010101p.pdf
- [6] M. Kiran *et al.*, “Enabling intent to configure scientific networks for high performance demands,” *Future Generation Computer Systems*, vol. 79, pp. 561–574, 2018.
- [7] M. Chen *et al.*, “Automatic deployment and control of network services in NFV environments,” *Journal of Network and Computer Applications*, vol. 167, 2020.
- [8] T. Kohler *et al.*, “A highly flexible and modular architecture for full-range distribution of event-based network control,” *IEEE Transactions on Network and Service Management*, vol. 15, no. 4, pp. 1392–1405, 2018.
- [9] W. Guan *et al.*, “A service-oriented deployment policy of end-to-end network slicing based on complex network theory,” *IEEE Access*, vol. 6, pp. 72403–72413, 2018.
- [10] P. Kokkinos *et al.*, “Pattern-driven resource allocation in optical networks,” *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 616–629, 2019.