

The Theoretical Starting Point and Progressive Research Agenda of MRAC: From Non-Convergence Diagnosis to Engineering AGI

— An L1–L5 Stratification of Propositions and Research Agenda for Multi-Round Audit Convergence

Ming Liu · ORCID: 0009-0001-2816-538X

Version v1.0 · August 2026

Abstract

Previous work introduced Multi-Round Audit Convergence (MRAC), which evaluates whether a system can stably handle a current task by tracking how issues change across repeated “audit–remediation–re-audit” cycles on the same engineering object. Another study explained uncertainty in AI delivery through the relationship among Search Space (S), Coverage Ability (C), and Delivery Reliability (R). Building on these two lines of prior work, this paper examines what kinds of coverage judgments can be supported by MRAC non-convergence and convergence, respectively, and organizes them into a progressive L1–L5 research agenda.

Based on explicit operational definitions and applicability conditions, this paper argues that, when task and audit conditions remain stable, identified issues are valid, remediations are continuously subjected to re-audit, and resource boundaries are specified in advance, persistent MRAC non-convergence can serve as a conditional diagnosis that the current AI development system as a whole cannot yet reliably cover the task search space. For L2, when remediation results can be preserved and drift is controlled, convergence indicates that exposed failure paths have been more thoroughly handled and validated, and provides positive evidence that local coverage under the current audit configuration is approaching saturation. The representativeness of this evidence for the full delivery-relevant search space, and the strength of any extrapolation from it, still require empirical validation. On this basis, L3 studies whether recursive decomposition can restore convergence for non-convergent tasks, L4 examines whether complete engineering workflows can advance autonomously over extended periods, and L5 tests whether similar mechanisms can extend to intelligent tasks beyond engineering.

Keywords: Multi-Round Audit Convergence; MRAC; SCR meta-relation; search space; coverage ability; delivery reliability; recursive decomposition and validation; engineering AGI; AI engineering

1. Introduction

Current AI systems can already generate large volumes of engineering output that appear highly complete, but a fundamental distinction remains between “being able to generate” and “being able to confirm that a result is ready for delivery.” A successful execution, a passing test, or a single audit round that finds no issues only indicates that no further failure has been exposed along the current observation path. It does not establish that the key delivery-relevant conditions and failure paths have been sufficiently covered. In long-running complex projects, the resulting risks accumulate over time: local omissions affect not only the current delivery but can also become embedded in the codebase, dependencies, interfaces, and project state, thereby expanding the search space that future tasks must handle.

Iterative feedback and correction are not unique to MRAC. Self-Refine repeatedly revises outputs through model-generated self-feedback, while Reflexion stores linguistic feedback as episodic memory to improve subsequent decisions [3–4]. Without external feedback, however, self-correction by models may be ineffective and can even reduce reasoning performance [5]; the performance of software-engineering agents can also be significantly affected by tool-interface design [6]. These studies primarily evaluate task performance, whereas MRAC asks a different question: whether multi-round issue trajectories can reflect the coverage relationship between a system and a task. Accordingly, this paper does not treat “multiple calls” themselves as reliable evidence, but instead analyzes the model, agent, tools, environment, audit process, and resource budget as a combined system.

An auditor likewise cannot be assumed to be an unbiased verifier. Research on LLM-as-a-judge has documented issues including position bias, verbosity preference, and self-enhancement bias [7]. Calboreanu’s case study of prompt specifications recorded non-monotonic defect convergence across nine audit rounds and included limited cross-model replication [8], providing relevant but indirect evidence that iterative audit trajectories can be empirically informative. That study, however, focused on quality assurance in prompt engineering. It did not formulate a conditional proposition linking non-convergence to insufficient reliable coverage, nor did it further address task decomposition or autonomous progression across the full engineering process.

This paper directly builds on prior work on MRAC and SCR. MRAC continuously audits, remediates, and re-audits an engineering object, observes whether newly identified valid issues decrease overall, and proposes the control logic of “proceed upon convergence; return to task decomposition upon non-convergence” [2]. The SCR meta-relation explains AI delivery outcomes in terms of the match among search space, coverage ability, and delivery reliability [1]. On this basis, the paper addresses three questions:

1. Under what conditions can MRAC non-convergence support the diagnosis that the current system cannot yet reliably cover the task search space?
2. To what extent can MRAC convergence support judgments of improved coverage and local coverage approaching saturation, and which extrapolations still require empirical validation?
3. Starting from these two theoretical foundations, how should the MRAC research program toward reliable delivery, engineering AGI, and broader AGI be progressively

organized?

These questions are organized into L1–L5. The “levels” here are neither a system-maturity score nor five conclusions claimed to be simultaneously established. Rather, they progress from conditional theoretical propositions to empirical validation, engineering mechanisms, a hypothesis of autonomous progression across the full engineering process, and a cross-domain research agenda.

2. Theoretical Premises and Unit of Analysis

2.1 Two Premises

Premise 1: MRAC is a second-order observation of the coverage process.

A single audit round produces only a first-order result: “what was found in this round.” Multiple audit rounds additionally reveal how newly identified issues change over time, whether remediations persist, whether severe issues continue to appear, and whether a stable state can be reproduced. This paper refers to such trajectory-based observation as a second-order observation. The object observed by MRAC is not the output of any single round, but the full audit–remediation sequence.

Premise 2: Reliable delivery depends on the match between search space and coverage ability.

Neither the task search space nor system coverage ability is a fixed scalar independent of the task. Task boundaries, engineering state, validation conditions, models, agents, tools, context organization, audit protocols, and resource budgets jointly determine whether the search space and coverage ability are matched. This paper does not require S, C, and R to be scalar quantities. It relies only on a directional premise: all else being equal, insufficient effective coverage of the delivery-relevant search space constrains the delivery reliability that can be achieved. Search-space governance changes task boundaries and the engineering environment so that a task can re-enter a range that the current system can handle stably with its existing coverage ability.

Describing problem solving in terms of search has a classical foundation in artificial intelligence [9]. However, the task search space in SCR is not equivalent to an enumerable classical state space; it is an analytical construct defined with respect to a specific delivery and jointly shaped by engineering state and governance methods. Likewise, delivery reliability in this paper is a system–task-level delivery judgment and does not replace established taxonomies of faults, errors, failures, and dependability attributes in software dependability research [11].

2.2 Unit of Analysis: Complete System and Task Session

The “system” discussed in this paper is not a single model invocation. It is the complete AI development system composed of:

- the foundation model and its reasoning settings;
- agents, retrieval, editing, and execution tools;
- context loading, the code repository, and the runtime environment;
- audit protocols, issue-severity classification, and remediation procedures;

- tests, runtime feedback, and other forms of external validation;
- state persistence, task decomposition, dependency management, merging, and roll-back mechanisms;
- the allowed time, token, compute, and invocation budgets.

The unit of analysis is a task session: from the point at which the system receives a relatively stable task intent to the point at which delivery is completed, the task is terminated, or the task is decomposed again. L1 evaluates the matching state of the “current system–current task–current resource boundary,” rather than the absolute capability of the foundation model outside its environment.

3. MRAC L1: Non-Convergence and Insufficient Reliable Coverage

3.1 Definitions Required for L1

To avoid circularly defining coverage status through audit results, L1 distinguishes four concepts.

Definition 1: Delivery-relevant substantive issue.

An issue is a delivery-relevant substantive issue if it affects key requirements, task boundaries, implementation validity, structural integrity, relevant state, regression risk, or the adequacy of validation, and may prevent the result from being trusted as a completed delivery. Wording preferences, formatting differences with no delivery impact, duplicate reports, and issues already shown to be invalid are excluded.

Definition 2: Reliable coverage.

Given a task, system conditions, a risk threshold, and a resource budget, the current task search space is considered to lie within the system’s reliably coverable range if the complete AI development system can stably identify the key conditions on which delivery depends, distinguish among important candidate paths, exclude decisive errors, resolve identified issues, and produce validation evidence commensurate with the risk. Reliable coverage does not require enumeration of every possibility. It does, however, require the system to stably handle the delivery-determining portion of the space through abstraction, elimination, feedback, and validation, rather than relying on a single accidental success.

Definition 3: Valid MRAC sequence.

An MRAC sequence may be used for an L1 judgment only if the following conditions hold:

1. Task goals, non-goals, core constraints, acceptance boundaries, and risk thresholds remain stable during the observation period.
2. The audit scope and core protocol remain comparable, and protocol extensions are recorded separately.
3. Each audit round is fully executed; rounds affected by tool failure, missing context, or abnormal interruption are excluded.
4. Newly identified issues are checked for validity and deduplicated, and are confirmed to be substantive issues in the current task.
5. Confirmed issues are remediated, revalidated, and included again in a full-scope audit. Merely modifying surface symptoms, deleting audit evidence, or narrowing the

scope does not count as valid remediation.

6. Changes in models, tools, context, and resource conditions are recorded sufficiently to determine the system boundary to which the current judgment applies.

Definition 4: Persistent non-convergence.

Within a pre-specified number of valid rounds, resource budget, or observation window, an MRAC sequence is persistently non-convergent if new medium- or high-severity issues, structural issues, or new issue categories continue to appear and the issue trajectory does not enter a stable region consistent with the risk threshold; or if remediated issues continue to recur, remediation-induced changes continue to generate new substantive issues, and the system cannot maintain a stable result. A task meeting any of these conditions is considered persistently non-convergent under the current conditions.

The requirement that the boundary be **pre-specified** is important. The number of rounds cannot be changed arbitrarily after observing the result, nor can an accidental clean round be selectively interpreted as convergence, or an unfinished long sequence as non-convergence.

3.2 The L1 Proposition

L1: Non-Convergence Diagnostic Proposition.

For a task satisfying the conditions of a valid MRAC sequence, if it remains persistently non-convergent within the current system and resource boundaries, then the current system's coverage ability is insufficient to reliably cover the task search space.

The phrase "cannot reliably cover" is conditional. It refers only to the combination of the current task boundary, engineering environment, model, agent, tools, context, audit and remediation protocols, and resource budget. It does not imply that the task is unsolvable in principle, nor that the same model would remain unable to solve it after system conditions change.

3.3 Argument

Given the definitions and premises above, the conditional diagnostic relationship in L1 can be shown by contradiction. Assume that a task satisfying the conditions of a valid MRAC sequence remains persistently non-convergent, while also assuming that the current system can reliably cover its search space within the specified resource boundary. By the definition of reliable coverage, the system should be able, within budget, to identify key conditions, exclude decisive errors, resolve identified issues, and complete validation, thereby compressing delivery-threatening uncertainty to within the risk threshold.

A valid MRAC sequence keeps task intent and audit scope stable and counts only genuine new issues that are relevant to delivery. If the system possesses the coverage ability described above, and if the valid MRAC sequence can continue to observe this coverage process, then as issues are identified and remediated and incorrect paths are eliminated, the audit trajectory should enter a stable region consistent with the risk threshold within the budget. The trajectory need not decrease monotonically and may expose deeper issues along the way. However, after the pre-specified boundary is reached, the trajectory should not continue to exhibit new medium- or high-severity or structural issues, nor should

previously resolved issues continue to recur.

Persistent non-convergence therefore indicates that at least one part of the coverage process has not reached stable completion: key conditions remain unidentified, important failure paths continue to escape detection, remediation results cannot be preserved, remediation repeatedly introduces new delivery risks, validation remains insufficient to support a stable judgment, or the required resources exceed the budget. This contradicts the assumption that the current system can reliably cover the task search space. Therefore, when the stated definitions and premises hold, persistent non-convergence can be used as a conditional diagnostic signal that the current system–task combination has not yet entered a reliably coverable region. The validity of L1 depends on these operational definitions and on the conditions of a valid MRAC sequence.

3.4 Theoretical Significance and Boundaries

Reliable coverage is independently defined by the system–task relationship above, while MRAC non-convergence is an observable process state. L1 does not directly define coverage status using audit results. Instead, it connects an indirectly observable system–task matching relationship to a recordable audit–remediation trajectory. An engineering system therefore does not need to possess a complete correct answer to the task in advance, nor does it need to fully quantify the search space, in order to identify its current coverage boundary through non-convergence.

Non-convergence may arise from task scale and boundaries, engineering state, models and agents, context, auditors, remediation processes, validation conditions, or resource budgets. L1 does not distinguish among these specific causes. It only determines that the current Coverage Ability (C) and Search Space (S) are not yet reliably matched, thereby triggering two classes of intervention: increasing coverage ability or governing the search space. What kinds of coverage judgments convergence can support, and the boundaries of extrapolation from those judgments, are addressed further in L2.

4. Preliminary Engineering Mapping of L1

The prior MRAC paper reported two internal software-engineering cases [2]. This paper does not repeat the full cases, but examines how their trajectories map to the L1 conditions.

Case	Key Trajectory	L1-Relevant Observation
A5PR1 Spec	P1 and P2 issues continued to appear from R2 through R14; R3 had one round with no new issues, but high-severity issues reappeared afterward	A single clean round did not establish stable coverage; the original task was stopped and further decomposed

Case	Key Trajectory	L1-Relevant Observation
B2PR-01 Spec and PR	The Spec was temporarily clean at R3; P1 appeared again at R6; after remediation, R7 and R8 were consecutively clean. After PR remediation, multiple consecutive rounds found no new issues	Accidental clean rounds must be distinguished from a reproducible stable region

The A5PR1 case shows that a single clean round cannot overturn a later trajectory of persistent non-convergence, nor is it sufficient to justify proceeding with the task. The B2PR-01 case indicates that a reproducible stable region provides more coverage evidence than a single-round result. Whether such evidence represents the full delivery-relevant search space, however, still requires independent validation and long-term outcome testing.

Future public experiments should hold the task baseline and audit conditions constant and retain original audit, remediation, deduplication, severity, and drift records, so as to test whether finite trajectories can be used consistently for L1 judgments.

5. The Progressive MRAC L1–L5 Structure

5.1 Overall Stratification

Level	Core Proposition	Nature of Proposition	Main Validation Requirement
L1 Non-Convergence Diagnosis	Persistent non-convergence indicates that the current system cannot reliably cover the task search space	Conditional diagnostic proposition	Establish the conditional structure; verify that real sequences satisfy the conditions
L2 Convergence and Coverage	Valid convergence indicates more thorough handling and validation of exposed failure paths, and provides positive evidence that local coverage under the current audit configuration is approaching saturation	Improvement in handling and validating exposed paths is directly supported by the process; local saturation and the scope of extrapolation require empirical validation	Determine the coverage range of different audit configurations, common blind spots, drift risk, delivery-reliability extrapolation, and audit ROI

Level	Core Proposition	Nature of Proposition	Main Validation Requirement
L3 Decomposition Recovery	Study whether non-convergent engineering tasks can regain convergence through search-space governance and recursive decomposition, thereby improving delivery reliability	Experimentally testable engineering mechanism	Paired cases, controlled experiments, independent delivery outcomes, and composition validation
L4 Autonomous Engineering Progression	Common engineering activities can jointly be transformed into a task graph that is auditable, remediable, decomposable, and composable	Core engineering-AGI hypothesis	Validate long-term autonomous progression of complete projects and its necessary conditions
L5 Cross-Domain Extension	MRAC-like recursive validation and governance mechanisms can extend to important intelligent tasks beyond engineering	Long-term AGI research agenda	Cross-domain validation of whether tasks can form explicit intermediate results, undergo audit and remediation, and continue to be decomposed

Each level has independent validation requirements. The establishment of a lower level does not automatically imply the establishment of higher levels.

6. L2: Convergence, Current Audit Coverage Range, and Empirical Validation

6.1 The L2 Proposition

L2: Convergence Coverage Judgment.

When task intent and the audit object remain stable, audits are effectively executed, identified issues are resolved, remediation results are preserved, and audit/remediation drift is controlled, MRAC convergence indicates that exposed failure paths have been handled and validated more thoroughly, and provides positive evidence that local coverage under the current audit configuration is approaching saturation. The extent to which this evidence represents the full delivery-relevant search space, and the strength of the delivery-reliability judgments it can support, must be continually validated against empirical evidence.

Here, the **current audit coverage range** refers to the delivery-relevant conditions and failure paths that a given configuration—under a specified model, agent, tools, context,

audit direction, audit prompt, validation method, and resource budget—can consistently access, distinguish, and validate across repeated and valid executions. It is only the portion of the full delivery-relevant search space that the current configuration may cover. It cannot be directly observed from a single convergent sequence and can only be estimated through repeated experiments, audits with different configurations, and evidence of later omissions.

6.2 What Theory Can Support and What Requires Empirical Extrapolation

By repeatedly discovering and resolving issues, MRAC continues to probe for failure paths that have not yet been exposed. If remediations pass validation and are not invalidated by subsequent modifications or drift, each valid cycle more thoroughly handles and validates the failure paths that have already been exposed. A convergent sequence can therefore directly support one relatively strong conclusion and provide positive evidence for another judgment.

First, **exposed paths have been handled and validated more thoroughly**. Compared with the initial object and intermediate non-convergent states, the system has identified, excluded, or validated a larger set of exposed delivery-relevant failure paths. This conclusion follows from the audit-remediation process itself and does not depend on observing successful delivery after the fact.

Second, **convergence provides positive evidence that local coverage is approaching saturation**. Under a pre-specified protocol, number of valid rounds, and risk threshold, if new substantive issues gradually decrease and then stably cease to appear, the current configuration finds it increasingly difficult, within that sequence, to expose additional major failure paths. This can serve as positive evidence that local coverage is approaching saturation. However, a finite convergent trajectory alone cannot strictly establish that the potentially discoverable paths are nearly exhausted. The strength of the evidence depends on the number of audit rounds, search diversity, configuration differences, and historical omission rates. Nor does this imply that the full delivery-relevant search space has been sufficiently covered or that the result must be correct.

6.3 Common Blind Spots and Coverage Expansion

Multi-round audits may share common blind spots. Even when the system converges within the observable range, it may still miss decisive delivery risks. The appearance of major omissions after convergence indicates a capability boundary in the current audit configuration. The more representative the current audit coverage range is of the full delivery-relevant search space, the stronger the delivery-reliability claims that convergence can support.

Using different models, audit directions, audit prompts, tools, and forms of external validation in combination can alter search paths and expand coverage. Diversity genuinely reduces common blind spots only when different methods can stably expose different types of issues. Repeated calls using homogeneous models and prompts may increase the number of rounds without significantly expanding coverage. Future research should determine which failure types different configurations can discover, how similar their search paths are, and how much newly added audit methods reduce post-convergence omissions.

6.4 Drift, Reliability Extrapolation, and Audit ROI

Audit and remediation drift determine whether a convergent sequence still corresponds to the same task. If remediation changes the original intent, narrows the acceptance scope, or breaks already resolved issues through side effects, the system may exhibit apparent convergence on a modified object. Theoretically, comparability of the task and audit object and preservation of remediation results are necessary conditions for L2. Engineering research must still examine the probability of drift, methods for detecting it, preventive mechanisms, and their costs.

Empirical research on L2 centers on four categories of questions: the coverage boundaries of different audit configurations; common blind spots and ways to reduce them; the occurrence and control of audit/remediation drift; and the predictive strength of convergence trajectories for actual delivery reliability. Hidden tests, differentiated independent audits, real-world operation, and post-delivery defects, rework, and rollbacks can all be used to validate these relationships.

Audit efficiency is also a core engineering issue. Audit ROI should account simultaneously for valid issue discovery, risk resolution, and reduction in eventual rework per unit of token, time, or compute resource. As rounds accumulate, marginal coverage gains may decline. Decisions about when to continue the original protocol, switch models or audit directions, stop auditing, or trigger decomposition under L1 need to be continuously optimized against trade-offs among coverage gains, delivery risk, and resource cost.

7. L3: Restoring Convergence Through Recursive Decomposition

L3: Decomposition-Recovery Hypothesis.

For a class of common engineering tasks that exceed the current reliably coverable range, adjusting task boundaries, isolating context, governing dependencies, and recursively decomposing the task may transform the original task into a set of subtasks within a convergent range. If both the subtasks and the integrated object achieve valid convergence and pass independent validation, overall delivery reliability may improve.

L3 advances L1 from a diagnostic mechanism to a recovery mechanism. L1 only establishes that the current combination is mismatched; L3 further investigates whether the search space can be actively changed so that the task re-enters the coverage range.

Modular decomposition is a classical governance method in software engineering, and decomposition criteria directly affect informational dependencies among modules and the cost of later modifications [10]. L3 does not reintroduce the principle of modularity. Instead, it adds a dynamic trigger-and-validation mechanism: non-convergence identifies that the current decomposition boundary has failed, while renewed convergence of subtasks and the integrated object tests whether the new boundary is effective.

A case capable of supporting L3 must contain the complete chain:

Original task does not converge → mismatch boundary is identified → task is re-decomposed or governed → subtasks converge separately → integrated object converges again → independent validation shows improved outcomes

Simply observing that “subtasks are easier to complete after decomposition” is not sufficient. Decomposition may transfer risk to dependencies, interfaces, or the integration stage. Even if all subtasks converge separately, the combined project state is not guaranteed to converge. Integration itself must therefore be treated as a new engineering object. If integration does not converge, the system should continue locating cross-boundary problems, adjust the decomposition, or roll back, rather than equating local stability of subtasks with overall reliability.

L3 can be validated through paired experiments on the same initial task: compare a path that continues remediation within the original boundary against a path that re-decomposes the task after reaching the non-convergence threshold, and evaluate governance benefits using final residual defects, integration outcomes, cost, and delivery time. A limited number of cases with complete, reproducible chains can establish that the mechanism exists in real engineering practice; its generality requires cumulative evidence across tasks and projects.

8. L4 and L5: From Autonomous Progression Across the Full Engineering Process to Cross-Domain Extension

8.1 L4: Autonomous Progression Across the Full Engineering Process

To clarify the scope of this paper, **engineering AGI** is defined here as the ability of AI, given only the necessary external goals, authorization, and real-world feedback, to complete complex projects that would otherwise require long-term collaboration by an ordinary human team, without humans carrying out the engineering process. Demonstrating only that code implementation can be recursively decomposed is insufficient. Complete engineering delivery also includes structuring intent, clarifying requirements, product and technical decision-making, task planning, implementation validation, integration and release, and long-term evolution. L4 investigates whether these activities can jointly be transformed into a task graph whose components are auditable, remediable, decomposable, and recomposable.

L4 does not assume that all implementation details are specified in advance. Rather, goals, constraints, preferences, decision authority, and feedback interfaces must be sufficient for the system to continue progressing, while temporarily missing information must be identifiable and convertible into retrieval, experimentation, clarification, or authorization tasks. For L4 to hold, five conditions must be satisfied: non-convergent tasks can continue to be decomposed or restructured; intermediate results and key decisions can be audited; subtasks can be recombined, and integration failures can be routed back into the workflow; state and evidence can persist across sessions; and missing intent information can be identified and explicitly assigned to humans or external systems for resolution.

If these conditions hold across common engineering activities, the system does not need to cover the entire project at once. It can continue progressing through local coverage, preservation of intermediate state, and composition-level audits. The object of validation for L4 is therefore the complete project: whether the system can autonomously advance a task graph over long periods while maintaining reliable delivery and continuously reducing human involvement in process-level activities such as task decomposition, state tracking,

and local remediation.

8.2 L5: Cross-Domain Extension

L5: Cross-Domain Extension Proposition.

For AI tasks that can be transformed into processes in which state persists, intermediate objects can be audited, remediations can be executed, feedback can be obtained, and recursive decomposition is possible, it is possible to investigate whether MRAC-like mechanisms can be established: use non-convergence to identify current coverage boundaries, use convergence to accumulate positive evidence, and continuously improve task success rates through recursive decomposition and validation. If such transformations can cover a sufficiently broad range of important intelligent activities, the MRAC research agenda could extend from engineering AGI toward broader AGI.

The key question for L5 is whether tasks outside engineering can likewise produce explicit and inspectable intermediate results, undergo audit and remediation, continue to be decomposed and recombined, and obtain effective feedback. Software engineering is comparatively well suited to these conditions. Research, strategy, creative work, and real-world action may instead face open-ended goals, delayed feedback, irreversible consequences, and value conflicts. Whether L5 holds depends on whether these domains can establish state-recording and validation methods that play equivalent functional roles.

9. Empirical Validation of Theoretical Propositions and Falsifiability of Higher-Level Hypotheses

The core empirical task is to determine whether real-world sequences satisfy the applicability conditions of L1 and L2, and the strength of the engineering judgments that finite trajectories can support. For L1, research must determine when a finite number of rounds is sufficient to judge “persistent non-convergence.” For L2, empirical relationships must be established among different audit configurations, convergence trajectories, and coverage claims. Both successful and failed cases should enter the evidence base so that capability boundaries can be identified.

L3–L5 make stronger claims about recovery, autonomous progression across the full engineering process, and cross-domain extension. Real-world evidence may weaken these claims, restrict their scope of applicability, or refute them. If decomposition fails to improve overall delivery, if engineering activities cannot form a complete process of sustained recursive progression, or if important non-engineering tasks cannot obtain the necessary state and feedback structures, the scope of the corresponding higher-level propositions must be narrowed or the propositions rejected.

10. Future Research and Engineering Directions

Near-term research should prioritize unified methods for trajectory recording and outcome validation, compare different models, audit directions, audit prompts, tools, and validation methods, and link convergence trajectories to actual delivery outcomes. On this basis, paired evidence for “non-convergence–decomposition–re-convergence” can be ac-

cumulated for L3, comparing how different governance methods affect search space, integration risk, cost, and delivery time.

The medium-term goal is to build a prototype for complete projects that supports long-term state management, recursive decomposition, composition-level audit, and rollback, transforming MRAC from an offline observation signal into an engineering control mechanism capable of driving task continuation or termination, audit-strategy switching, decomposition, and rollback. Only after sufficient evidence for autonomous progression across the full engineering process has been accumulated does it become necessary to systematically test the applicability conditions of L5 in other domains. Specific infrastructure architectures are beyond the scope of this paper.

11. Conclusion

This paper establishes the theoretical starting point of MRAC and a progressive L1–L5 research agenda, distinguishing claims at different levels by evidential strength. L1 provides a conditional diagnosis; L2 provides positive coverage evidence within a limited scope; and L3–L5 progressively move into decomposition recovery, autonomous progression across the full engineering process, and cross-domain extension, all of which require experimental and long-term evidence. MRAC therefore does not need to assume the validity of its higher-level hypotheses in advance. It can instead begin from observable trajectories and progressively advance research and engineering validation across distinct levels of evidence.

References

- [1] LIU M. From the SCR Meta-Relation to Search-Space Governance: A Foundational Mechanism for Delivery Reliability in AI Software Engineering [EB/OL]. Zenodo, 2026. DOI: [10.5281/zenodo.21902420](https://doi.org/10.5281/zenodo.21902420).
- [2] LIU M. Multi-Round Audit Convergence: An External Validation Mechanism and Recursive Task Loop for Engineering AGI [EB/OL]. Zenodo, 2026. DOI: [10.5281/zenodo.21804751](https://doi.org/10.5281/zenodo.21804751).
- [3] MADAAN A, TANDON N, GUPTA P, et al. Self-Refine: Iterative refinement with self-feedback [C]//Advances in Neural Information Processing Systems 36. 2023. DOI: [10.52202/075280-2019](https://doi.org/10.52202/075280-2019).
- [4] SHINN N, CASSANO F, GOPINATH A, NARASIMHAN K, YAO S. Reflexion: Language agents with verbal reinforcement learning [C]//Advances in Neural Information Processing Systems 36. 2023. DOI: [10.52202/075280-0377](https://doi.org/10.52202/075280-0377).
- [5] HUANG J, CHEN X, MISHRA S, et al. Large language models cannot self-correct reasoning yet [C]//The Twelfth International Conference on Learning Representations. 2024. URL: <https://openreview.net/forum?id=Ikmd3fKBPQ>.
- [6] YANG J, JIMENEZ C E, WETTIG A, et al. SWE-agent: Agent–computer interfaces enable automated software engineering [C]//Advances in Neural Information Processing Systems 37. 2024. DOI: [10.52202/079017-1601](https://doi.org/10.52202/079017-1601).
- [7] ZHENG L, CHIANG W L, SHENG Y, et al. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena [C]//Advances in Neural Information Processing Systems 36. 2023: 46595–

46623. DOI: [10.52202/075280-2020](https://doi.org/10.52202/075280-2020).

[8] CALBOREANU E. Iterative audit convergence in LLM-managed multi-agent systems: A case study in prompt-engineering quality assurance [J]. *Software*, 2026, 5(2): 26. DOI: [10.3390/software5020026](https://doi.org/10.3390/software5020026).

[9] NEWELL A, SIMON H A. Computer science as empirical inquiry: Symbols and search [J]. *Communications of the ACM*, 1976, 19(3): 113–126. DOI: [10.1145/360018.360022](https://doi.org/10.1145/360018.360022).

[10] PARNAS D L. On the criteria to be used in decomposing systems into modules [J]. *Communications of the ACM*, 1972, 15(12): 1053–1058. DOI: [10.1145/361598.361623](https://doi.org/10.1145/361598.361623).

[11] AVIŽIENIS A, LAPRIE J C, RANDELL B, LANDWEHR C. Basic concepts and taxonomy of dependable and secure computing [J]. *IEEE Transactions on Dependable and Secure Computing*, 2004, 1(1): 11–33. DOI: [10.1109/TDSC.2004.2](https://doi.org/10.1109/TDSC.2004.2).