

The Agent Reliability Loop

Evals, Observability, Testing, and Production Operations

Ngo Trung Kien

Contents

Copyright and publication notice	1
Introduction: the promise and the boundary	3
Visual frontispiece	4
0.1 Who this book is for	4
0.2 Thesis and falsification condition	4
1 Define Reliability for a Trajectory	7
1.1 In plain terms	7
1.1.1 Chapter contract	7
1.2 The decision at stake	7
1.3 Working model	8
1.3.1 See the chapter in practice	8
1.4 Technical deep dive	10
1.4.1 Build sequence	10
1.4.2 Evidence to retain	10
1.4.3 Tests that can disprove the design	10
1.4.4 Design trade-off	10
1.4.5 Chapter-specific operating protocol	10
1.4.6 Operating workflow	13
1.4.7 Evidence and control graph	14
1.5 Worked composite case file — illustrative, not empirical evidence	15
1.5.1 Scenario trace	16
1.5.2 Artifact handoff	17
1.5.3 Evidence application	17
1.5.4 Transfer test	17
1.6 Measurement and recovery decision	17
1.7 Practice lab	18
1.8 Limits and judgment	18
1.9 Takeaways	19
1.10 Reflection	19
2 Build a Failure Taxonomy	21
2.1 In plain terms	21
2.1.1 Chapter contract	21
2.2 The decision at stake	21
2.3 Working model	22
2.3.1 See the chapter in practice	22
2.4 Technical deep dive	24
2.4.1 Build sequence	24
2.4.2 Evidence to retain	24

2.4.3	Tests that can disprove the design	24
2.4.4	Design trade-off	24
2.4.5	Chapter-specific operating protocol	24
2.4.6	Operating workflow	27
2.4.7	Evidence and control graph	28
2.5	Worked composite case file — illustrative, not empirical evidence	29
2.5.1	Scenario trace	30
2.5.2	Artifact handoff	31
2.5.3	Evidence application	31
2.5.4	Transfer test	31
2.6	Measurement and recovery decision	31
2.7	Practice lab	32
2.8	Limits and judgment	32
2.9	Takeaways	33
2.10	Reflection	33
3	Design Representative Eval Tasks	35
3.1	In plain terms	35
3.1.1	Chapter contract	35
3.2	The decision at stake	35
3.3	Working model	36
3.3.1	See the chapter in practice	36
3.4	Technical deep dive	38
3.4.1	Build sequence	38
3.4.2	Evidence to retain	38
3.4.3	Tests that can disprove the design	38
3.4.4	Design trade-off	38
3.4.5	Chapter-specific operating protocol	38
3.4.6	Operating workflow	41
3.4.7	Evidence and control graph	42
3.5	Worked composite case file — illustrative, not empirical evidence	43
3.5.1	Scenario trace	44
3.5.2	Artifact handoff	45
3.5.3	Evidence application	45
3.5.4	Transfer test	45
3.6	Measurement and recovery decision	45
3.7	Practice lab	46
3.8	Limits and judgment	46
3.9	Takeaways	47
3.10	Reflection	47
4	Grade Outcomes and Paths	49
4.1	In plain terms	49
4.1.1	Chapter contract	49
4.2	The decision at stake	49
4.3	Working model	50

4.3.1	See the chapter in practice	50
4.4	Technical deep dive	52
4.4.1	Build sequence	52
4.4.2	Evidence to retain	52
4.4.3	Tests that can disprove the design	52
4.4.4	Design trade-off	52
4.4.5	Chapter-specific operating protocol	52
4.4.6	Operating workflow	55
4.4.7	Evidence and control graph	56
4.5	Worked composite case file — illustrative, not empirical evidence	57
4.5.1	Scenario trace	58
4.5.2	Artifact handoff	59
4.5.3	Evidence application	59
4.5.4	Transfer test	59
4.6	Measurement and recovery decision	59
4.7	Practice lab	60
4.8	Limits and judgment	60
4.9	Takeaways	61
4.10	Reflection	61
5	Calibrate Automated Judges	63
5.1	In plain terms	63
5.1.1	Chapter contract	63
5.2	The decision at stake	63
5.3	Working model	64
5.3.1	See the chapter in practice	64
5.4	Technical deep dive	66
5.4.1	Build sequence	66
5.4.2	Evidence to retain	66
5.4.3	Tests that can disprove the design	66
5.4.4	Design trade-off	66
5.5	Judge-calibration calculation	66
5.5.1	Chapter-specific operating protocol	67
5.5.2	Operating workflow	69
5.5.3	Evidence and control graph	70
5.6	Worked composite case file — illustrative, not empirical evidence	71
5.6.1	Scenario trace	72
5.6.2	Artifact handoff	73
5.6.3	Evidence application	73
5.6.4	Transfer test	73
5.7	Measurement and recovery decision	73
5.8	Practice lab	74
5.9	Limits and judgment	74
5.10	Takeaways	75
5.11	Reflection	75

6	Test Tools and Environment State	77
6.1	In plain terms	77
6.1.1	Chapter contract	77
6.2	The decision at stake	77
6.3	Working model	78
6.3.1	See the chapter in practice	78
6.4	Technical deep dive	80
6.4.1	Build sequence	80
6.4.2	Evidence to retain	80
6.4.3	Tests that can disprove the design	80
6.4.4	Design trade-off	80
6.4.5	Chapter-specific operating protocol	80
6.4.6	Operating workflow	83
6.4.7	Evidence and control graph	84
6.5	Worked composite case file — illustrative, not empirical evidence	85
6.5.1	Scenario trace	86
6.5.2	Artifact handoff	87
6.5.3	Evidence application	87
6.5.4	Transfer test	87
6.6	Measurement and recovery decision	87
6.7	Practice lab	88
6.8	Limits and judgment	88
6.9	Takeaways	89
6.10	Reflection	89
	Midpoint visual synthesis	90
7	Run Adversarial and Perturbation Tests	91
7.1	In plain terms	91
7.1.1	Chapter contract	91
7.2	The decision at stake	91
7.3	Working model	92
7.3.1	See the chapter in practice	92
7.4	Technical deep dive	94
7.4.1	Build sequence	94
7.4.2	Evidence to retain	94
7.4.3	Tests that can disprove the design	94
7.4.4	Design trade-off	94
7.4.5	Chapter-specific operating protocol	94
7.4.6	Operating workflow	97
7.4.7	Evidence and control graph	98
7.5	Worked composite case file — illustrative, not empirical evidence	99
7.5.1	Scenario trace	100
7.5.2	Artifact handoff	101
7.5.3	Evidence application	101
7.5.4	Transfer test	101
7.6	Measurement and recovery decision	101

7.7	Practice lab	102
7.8	Limits and judgment	102
7.9	Takeaways	103
7.10	Reflection	103
8	Observe Traces in Production	105
8.1	In plain terms	105
8.1.1	Chapter contract	105
8.2	The decision at stake	105
8.3	Working model	106
8.3.1	See the chapter in practice	106
8.4	Technical deep dive	108
8.4.1	Build sequence	108
8.4.2	Evidence to retain	108
8.4.3	Tests that can disprove the design	108
8.4.4	Design trade-off	108
8.4.5	Chapter-specific operating protocol	108
8.4.6	Operating workflow	111
8.4.7	Evidence and control graph	112
8.5	Worked composite case file — illustrative, not empirical evidence	113
8.5.1	Scenario trace	114
8.5.2	Artifact handoff	115
8.5.3	Evidence application	115
8.5.4	Transfer test	115
8.6	Measurement and recovery decision	115
8.7	Practice lab	116
8.8	Limits and judgment	116
8.9	Takeaways	117
8.10	Reflection	117
9	Detect Drift and Quiet Failure	119
9.1	In plain terms	119
9.1.1	Chapter contract	119
9.2	The decision at stake	119
9.3	Working model	120
9.3.1	See the chapter in practice	120
9.4	Technical deep dive	122
9.4.1	Build sequence	122
9.4.2	Evidence to retain	122
9.4.3	Tests that can disprove the design	122
9.4.4	Design trade-off	122
9.4.5	Chapter-specific operating protocol	122
9.4.6	Operating workflow	125
9.4.7	Evidence and control graph	126
9.5	Worked composite case file — illustrative, not empirical evidence	127
9.5.1	Scenario trace	128

9.5.2	Artifact handoff	129
9.5.3	Evidence application	129
9.5.4	Transfer test	129
9.6	Measurement and recovery decision	129
9.7	Practice lab	130
9.8	Limits and judgment	130
9.9	Takeaways	131
9.10	Reflection	131
10	Gate Releases and Roll Back	133
10.1	In plain terms	133
10.1.1	Chapter contract	133
10.2	The decision at stake	133
10.3	Working model	134
10.3.1	See the chapter in practice	134
10.4	Technical deep dive	136
10.4.1	Build sequence	136
10.4.2	Evidence to retain	136
10.4.3	Tests that can disprove the design	136
10.4.4	Design trade-off	136
10.4.5	Chapter-specific operating protocol	136
10.4.6	Operating workflow	139
10.4.7	Evidence and control graph	140
10.5	Worked composite case file — illustrative, not empirical evidence	141
10.5.1	Scenario trace	142
10.5.2	Artifact handoff	143
10.5.3	Evidence application	143
10.5.4	Transfer test	143
10.6	Measurement and recovery decision	143
10.7	Practice lab	144
10.8	Limits and judgment	144
10.9	Takeaways	145
10.10	Reflection	145
11	Investigate Agent Incidents	147
11.1	In plain terms	147
11.1.1	Chapter contract	147
11.2	The decision at stake	147
11.3	Working model	148
11.3.1	See the chapter in practice	148
11.4	Technical deep dive	150
11.4.1	Build sequence	150
11.4.2	Evidence to retain	150
11.4.3	Tests that can disprove the design	150
11.4.4	Design trade-off	150
11.4.5	Chapter-specific operating protocol	150

11.4.6	Operating workflow	153
11.4.7	Evidence and control graph	154
11.5	Worked composite case file — illustrative, not empirical evidence	155
11.5.1	Scenario trace	156
11.5.2	Artifact handoff	157
11.5.3	Evidence application	157
11.5.4	Transfer test	157
11.6	Measurement and recovery decision	157
11.7	Practice lab	158
11.8	Limits and judgment	158
11.9	Takeaways	159
11.10	Reflection	159
12	Close the Learning Loop	161
12.1	In plain terms	161
12.1.1	Chapter contract	161
12.2	The decision at stake	161
12.3	Working model	162
12.3.1	See the chapter in practice	162
12.4	Technical deep dive	164
12.4.1	Build sequence	164
12.4.2	Evidence to retain	164
12.4.3	Tests that can disprove the design	164
12.4.4	Design trade-off	164
12.4.5	Chapter-specific operating protocol	164
12.4.6	Operating workflow	167
12.4.7	Evidence and control graph	168
12.5	Worked composite case file — illustrative, not empirical evidence	169
12.5.1	Scenario trace	170
12.5.2	Artifact handoff	171
12.5.3	Evidence application	171
12.5.4	Transfer test	171
12.6	Measurement and recovery decision	171
12.7	Practice lab	172
12.8	Limits and judgment	172
12.9	Takeaways	173
12.10	Reflection	173
	Closing: keep the evidence visible	175
	A compact implementation workbook	177
	Decision register	178
	Evidence scorecard	179
	Exception log	180
	Ninety-day plan	181
	Control and test matrix	182

Workflow map	183
13 References	185

Define Reliability for a Trajectory

In plain terms

Reliability joins outcome correctness path safety constraint compliance recovery and repeatability under stated conditions. In an agent evaluation and production-reliability loop, the practical question is **What does reliable completion mean for a variable multi-step task?** For ML engineers, platform teams, quality engineers, SREs, and product owners responsible for agents in production, the answer counts only when **Write a task specification with terminal states trajectory constraints and service limits** produces reviewable evidence for **constraint-adjusted task success rate** without normalizing **counting the final answer while ignoring the path used to obtain it**.

Chapter contract

DECIDE

What does reliable completion mean for a variable multi-step task?

BUILD

Write a task specification with terminal states trajectory constraints and service limits.

MEASURE

constraint-adjusted task success rate

STOP OR REDESIGN WHEN

counting the final answer while ignoring the path used to obtain it

The decision at stake

For **Define Reliability for a Trajectory**, the reliability owner must decide **what does reliable completion mean for a variable multi-step task?** before the evaluated task crosses from discussion into consequence for evaluators, operators, and affected users. The minimum decision record is the artifact **Write a task specification with terminal states trajectory constraints and service limits**, the observable **constraint-adjusted task success rate**, and the explicit hold condition **counting the final answer while ignoring the path used to obtain it**. If any Chapter 1 field is missing, the safe default is to contain the failing trajectory and roll

back to the verified release.

The bounded **Define Reliability for a Trajectory** decision advances the book's promise to build a closed reliability system that converts failures into eval cases, release gates, operational signals, and safer product changes. It does not endorse a product or predetermine the Chapter 1 answer; it creates a record that another person can challenge for each evaluation batch, deployment, drift review, and incident.

Working model

Read the **Define Reliability for a Trajectory** model as four linked fields. **Outcome:** Reliability joins outcome correctness path safety constraint compliance recovery and repeatability under stated conditions. **Boundary:** the design must make **counting the final answer while ignoring the path used to obtain it** visible rather than convenient. **Evidence:** interpret **constraint-adjusted task success rate** together with review effort and excluded cases. **Authority:** the reliability owner may continue, narrow, escalate, or stop the evaluated task. For **Define Reliability for a Trajectory**, Reference [1] from Anthropic, *Demystifying evals for AI agents*, informs a Chapter 1 boundary; it is not evidence that the local method or illustrative target will work.

See the chapter in practice

The **define reliability for a trajectory in practice** maps *The Agent Reliability Loop* Chapter 1 into a field scene before its abstract models begin. In one frame, it holds the question **What does reliable completion mean for a variable multi-step task?**, the artifact **Write a task specification with terminal states trajectory constraints and service limits.**, the proof signal **constraint-adjusted task success rate**, and the warning **counting the final answer while ignoring the path used to obtain it**.

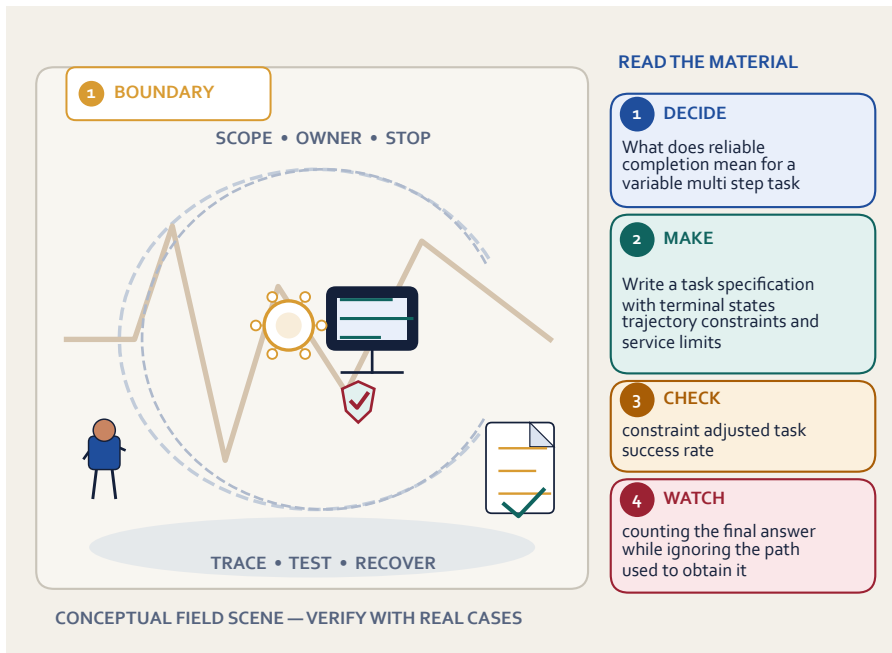


Figure 1.1: Chapter 1 field scene for Define Reliability for a Trajectory: a boundary view in the book's reliability setting.

Read the scene: For The Agent Reliability Loop, Chapter 1, begin with the central reliability scene and scan DECIDE, MAKE, CHECK, and WATCH. Treat the Define Reliability for a Trajectory scene as conceptual, then use its cases and evidence records to test whether constraint-adjusted task success rate actually improves. For **Define Reliability for a Trajectory**, this explanatory plate is a reasoning prompt—not observed evidence or a universal operating prescription.

Visual field note — Define Reliability for a Trajectory:

- 1. Locate the tension.** In this composite situation, A browsing agent finds the right price after sending private data to an unapproved site.
- 2. Name the choice.** Answer the chapter question: **What does reliable completion mean for a variable multi-step task?**
- 3. Produce the record.** Use **Write a task specification with terminal states trajectory constraints and service limits.**, then retain the reliability owner, assumptions, and case evidence needed to review it.
- 4. Test and return.** Track **constraint-adjusted task success rate.** If **counting the final answer while ignoring the path used to obtain it**, return to DECIDE and revise the evaluation boundary instead of explaining the warning away.

Technical deep dive

For agents, reliability is conditional: given a task distribution, tool environment, policy set, budgets, and time window, the production agent reaches acceptable terminal states through permitted trajectories. Report the denominator and conditions with every rate.

Build sequence

1. Define terminal states and unacceptable path events.
2. Specify the task and environment distribution.
3. Measure repeat runs where stochastic variance matters.
4. Report confidence intervals and severe failures separately.

Evidence to retain

- Reliability definition sheet
- Trajectory constraint catalogue
- Segmented result table

Tests that can disprove the design

- Correct outcome with prohibited tool
- Safe refusal on impossible task
- Same task repeated under controlled randomness

Design trade-off

A strict definition may lower headline success while making real risk visible. Within *The Agent Reliability Loop*, record the rationale for this trade-off, the signal that would reverse it, and the person authorized to decide.

Chapter-specific operating protocol

Phase	Reader action	Evidence retained	Hold condition
1. Define	Write the exact decision behind What does reliable completion mean for a variable multi-step task?	Chapter 1 owner, input, output, exception, and timestamp.	Hold Define Reliability for a Trajectory if its record cannot be reconstructed.
2. Contrast	Apply Write a task specification with terminal states trajectory constraints and service limits to one ordinary case.	Chapter 1 owner, input, output, exception, and timestamp.	Hold Define Reliability for a Trajectory if its record cannot be reconstructed.
3. Exclude	Run the same artifact against a counter case shaped by counting the final answer while ignoring the path used to obtain it.	Chapter 1 owner, input, output, exception, and timestamp.	Hold Define Reliability for a Trajectory if its record cannot be reconstructed.
4. Assign	Name the reliability owner as the person who may continue, narrow, or stop.	Chapter 1 owner, input, output, exception, and timestamp.	Hold Define Reliability for a Trajectory if its record cannot be reconstructed.
5. Challenge	Record constraint-adjusted task success rate and the review effort for the same case.	Chapter 1 owner, input, output, exception, and timestamp.	Hold Define Reliability for a Trajectory if its record cannot be reconstructed.

Phase	Reader action	Evidence retained	Hold condition
6. Publish	Release the Define Reliability for a Trajectory artifact only when its evidence answers this chapter’s question.	Chapter 1 owner, input, output, exception, and timestamp.	Hold Define Reliability for a Trajectory if its record cannot be reconstructed.

This sequence is specific to **Define Reliability for a Trajectory**. For that Chapter 1 decision, it is not permission to automate; it is a way for ML engineers, platform teams, quality engineers, SREs, and product owners responsible for agents in production to make the reasoning inspectable for each evaluation batch, deployment, drift review, and incident.

Write a task specification with terminal states trajectory constraints and service limits. This is the chapter’s core production artifact. The **Define Reliability for a Trajectory** artifact should fit on one page, use ordinary language, and be reviewed by someone who did not design it. A compact artifact exposes disagreements that a long narrative can conceal.

The chapter’s visual tool is the **chapter contract**. For **Define Reliability for a Trajectory**, it translates explanation into an inspectable decision aid: the reader can see the decision, artifact, evidence, and exact condition that pauses the evaluated task.

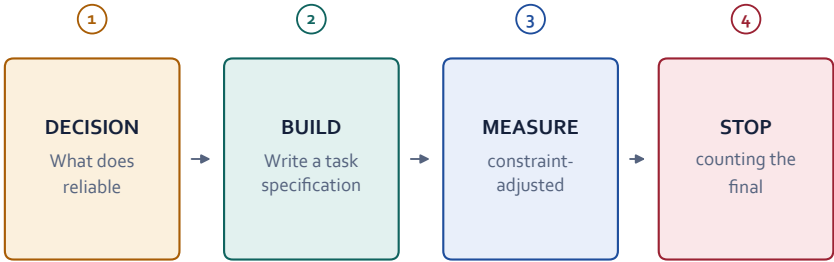


Figure 1.2: Chapter contract connecting the decision, artifact, measure, and stop condition for Define Reliability for a Trajectory.

How to use this page: Move across the four Define Reliability for a Trajectory cards; compare the decision with constraint-adjusted task success rate and stop before counting the final answer while ignoring the path used to obtain it. The chapter-specific measure is **constraint-adjusted task success rate**; the principal warning is **counting the final answer while ignoring the path used to obtain it**.

Operating workflow

The **define reliability for a trajectory workflow** makes the chapter executable. For **Define Reliability for a Trajectory**, the five stages carry **Write a task specification with terminal states trajectory constraints and service limits**. from the question **What does reliable completion mean for a variable multi-step task?** to a test against **constraint-adjusted task success rate**, with **counting the final answer while ignoring the path used to obtain it** as an explicit return or stop signal.

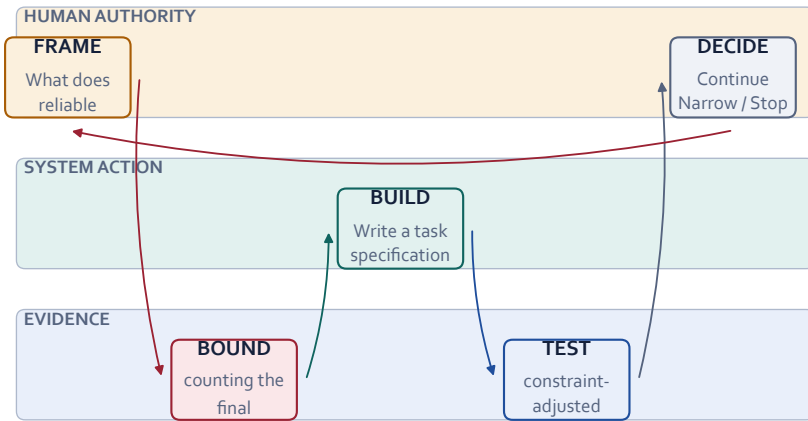


Figure 1.3: Five-stage workflow for chapter 1, Define Reliability for a Trajectory: frame the decision, set the evaluation boundary, build Write a task specification with terminal states trajectory constraints and service limits., test against constraint-adjusted task success rate, then continue, narrow, or stop before counting the final answer while ignoring the path used to obtain it.

Read the workflow: Move from FRAME to DECIDE for Define Reliability for a Trajectory. The lower return path is mandatory when evidence is missing or the stop signal—counting the final answer while ignoring the path used to obtain it—appears. For **Define Reliability for a Trajectory**, record the reliability owner and evidence at every handoff; an arrow is not evidence that the preceding stage passed.

Evidence and control graph

The **define reliability for a trajectory — evidence network** shows the relationships that must remain inspectable when the workflow runs. In *The Agent Reliability Loop*, this qualitative graph asks the reader to record observed evidence for **constraint-adjusted task success rate** and route **counting the final answer while ignoring the path used to obtain it** to recovery instead of treating sample numbers as facts.

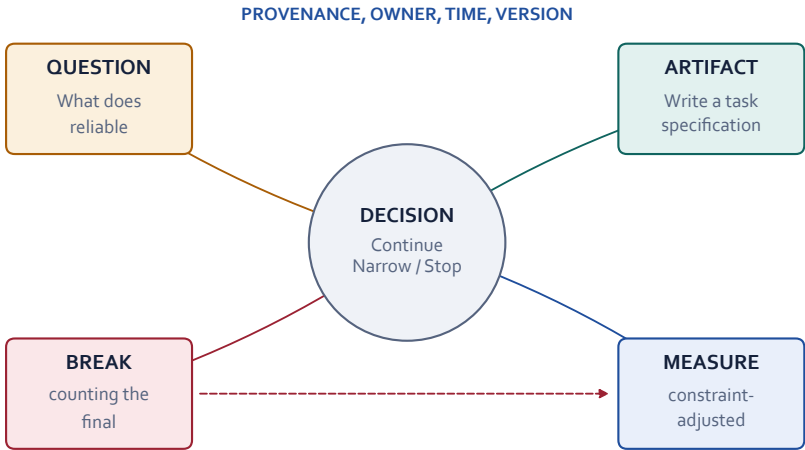


Figure 1.4: Qualitative evidence network for chapter 1, Define Reliability for a Trajectory, linking the question to Write a task specification with terminal states trajectory constraints and service limits., the measure constraint-adjusted task success rate, and the recovery condition counting the final answer while ignoring the path used to obtain it without invented empirical values.

Read the graph: Trace the evidence for Define Reliability for a Trajectory toward the decision. For Define Reliability for a Trajectory, empty or weak evidence does not count as zero risk; it routes the evaluated task to verification, escalation, or stop. Use this **Define Reliability for a Trajectory** graph beside **Write a task specification with terminal states trajectory constraints and service limits**, not as a substitute for recorded cases.

Worked composite case file — illustrative, not empirical evidence

This **Define Reliability for a Trajectory** record turns the scenario into an inspectable rehearsal in an agent evaluation and production-reliability loop. Its numbers are worksheet examples for Chapter 1, not observed results or a prediction of benefit.

Record field	Filled illustrative entry
Situation	A browsing agent finds the right price after sending private data to an unapproved site.
People affected	evaluators, operators, and affected users
Decision	What does reliable completion mean for a variable multi-step task?
Artifact excerpt	Write a task specification with terminal states trajectory constraints and service limits.
Accountable authority	The reliability owner.
Measure and unit	constraint-adjusted task success rate — eligible cases satisfying the rule divided by all eligible cases.
Example baseline	59/100.
Example target / decision rule	at least 75/100; move only toward the stated target with exceptions reported separately.
Named exception	counting the final answer while ignoring the path used to obtain it.
Safe default	Contain the failing trajectory and roll back to the verified release.

Scenario trace

1. **Preserve the starting state.** For **Define Reliability for a Trajectory**, record the original request, governing rule, available evidence, and the last safe result before changing anything.
2. **Make the chapter decision.** Answer **What does reliable completion mean for a variable multi-step task?** and record the Chapter 1 assumption that carries the most consequence for evaluators, operators, and affected users.
3. **Use the artifact.** Complete **Write a task specification with terminal states trajectory constraints and service limits** with a normal case and a counter case; in this chapter, do not infer missing facts merely to finish the form.
4. **Observe the decision rule.** Review **constraint-adjusted task success rate** for each evaluation batch, deployment, drift review, and incident; the **Define Reliability for a Trajectory** record reports exclusions and review effort beside the headline value.

5. **Contain and recover.** If **counting the final answer while ignoring the path used to obtain it**, contain the failing trajectory and roll back to the verified release. Preserve the Chapter 1 failed case, assign a deadline, and require the reliability owner to authorize any restart.

Artifact handoff

Start **Define Reliability for a Trajectory** from the book-level thesis and boundary. Within *The Agent Reliability Loop*, attach the current artifact and exception record, then hand both to Chapter 2, **Build a Failure Taxonomy**, which asks: How should teams name failures consistently enough to learn from them?. That recipient may revise the **Define Reliability for a Trajectory** assumption, but must preserve the evidence that made this book's earlier decision reasonable.

Evidence application

Reference [1], Anthropic, *Demystifying evals for AI agents*, provides a relevant boundary or design input for **Define Reliability for a Trajectory**. It does **not** prove that the Chapter 1 worked values, local method, or target will succeed. The **Define Reliability for a Trajectory** reviewer must link this book's exact claim to the exact source passage and leave unsupported synthesis marked UNVERIFIED.

Transfer test

- Re-run **Write a task specification with terminal states trajectory constraints and service limits** with one routine Chapter 1 case that should pass.
- Re-run the **Define Reliability for a Trajectory** artifact with missing evidence and keep the safe default active.
- Re-run it where **counting the final answer while ignoring the path used to obtain it** appears plausible but not certain.
- Ask a reviewer who did not design this chapter's method to decide whether the retained record supports continuing, narrowing, escalating, or stopping.

Measurement and recovery decision

For **Define Reliability for a Trajectory**, define **constraint-adjusted task success rate** as eligible cases satisfying the rule divided by all eligible cases. In this **Define Reliability for a Trajectory** worksheet, the Chapter 1 worked baseline is 59/100 and the illustrative target is at least 75/100; both must be replaced with evidence from an agent evaluation

and production-reliability loop. Review the Chapter 1 result for each evaluation batch, deployment, drift review, and incident, preserve excluded and high-consequence cases, and interpret movement as better only when it is toward the stated target with exceptions reported separately.

The named failure is **counting the final answer while ignoring the path used to obtain it**. Containment means: contain the failing trajectory and roll back to the verified release. Recovery is complete only after the reliability owner can reconstruct the failed case, explain which assumption changed, and authorize a new test against **constraint-adjusted task success rate**.

Practice lab

Choose one live but reversible **Define Reliability for a Trajectory** example from an agent evaluation and production-reliability loop. Create the following:

- a one-sentence answer to **What does reliable completion mean for a variable multi-step task?**;
- a completed first draft of **Write a task specification with terminal states trajectory constraints and service limits**;
- one routine case, one missing-evidence case, and one case shaped by **counting the final answer while ignoring the path used to obtain it**;
- a locally measured baseline for **constraint-adjusted task success rate** using the uniteligible cases satisfying the rule divided by all eligible cases;
- a stop threshold, the reliability owner as owner, and the safe default contain the failing trajectory and roll back to the verified release;
- reliability evidence packet that will be handed to the next chapter.

Ask a colleague to challenge **Write a task specification with terminal states trajectory constraints and service limits**. They should identify one ambiguous field, one person among evaluators, operators, and affected users whose interests are missing, and one reason **constraint-adjusted task success rate** could reward the wrong behavior. Preserve that challenge in the chapter handoff before revising.

Limits and judgment

Define Reliability for a Trajectory supplies a locally testable method, not proof of causal effectiveness or a universal rule. This Chapter 1 method remains inside the book boundary: This book does not certify any agent as safe, replace statistical or domain expertise, or treat benchmark success as production validity. Readers applying **Define Reliability for a**

Trajectory for evaluators, operators, and affected users must verify current law, contract, policy, safeguarding, accessibility, security, and professional duties before consequential use. A negative **constraint-adjusted task success rate** result may mean the question was framed badly, the review cost is too high, or contain the failing trajectory and roll back to the verified release is the responsible decision.

Takeaways

- Reliability joins outcome correctness path safety constraint compliance recovery and repeatability under stated conditions.
- The chapter artifact is: write a task specification with terminal states trajectory constraints and service limits.
- The primary measure is **constraint-adjusted task success rate**, paired with review cost and failure severity.
- Watch especially for **counting the final answer while ignoring the path used to obtain it**.
- In an agent evaluation and production-reliability loop, expand scope only after the reliability owner can show the evidence and exercise the recovery path.

Reflection

What evidence would let a skeptical reviewer answer **What does reliable completion mean for a variable multi-step task?** differently from the reliability owner? Which case involving evaluators, operators, and affected users would **constraint-adjusted task success rate** miss, and would that omission require continuing, narrowing, escalating, or contain the failing trajectory and roll back to the verified release?

