TOOLKIT — FIELD GUIDE

The RAID Log.

Why yours died in week six, what the four letters are actually for, and the one number that says whether the thing predicts anything.

Companion to the RAID Log workbook.
Read this once. Live in the log.

CNNCTD, LLC — Working. Better.

cnnctd.work

READ THIS FIRST

You already know what a RAID log is. That has never been the problem.

There is no shortage of templates. There is a considerable shortage of RAID logs that are still true in week ten.

So this guide skips the part where somebody explains that R stands for Risks. What it covers is the thing every template leaves out: RAID logs do not fail because people do not understand the categories. They fail in four specific, predictable ways, and every one of them is visible in the file before it becomes visible in the project.

The four ways it dies

- **The register only grows.** Nothing ever gets closed, because most entries were written in a form that cannot be closed. Add twenty items a month, close none, and by month three nobody opens it.
- **It becomes a compliance artifact.** Maintained because a steering committee asks for it, read by nobody, and updated in the fifteen minutes before the meeting. Technically current. Functionally decorative.
- **Everything drifts into Risks.** Assumptions and dependencies get logged as risks because that tab already exists, and then they inherit the risk review cadence — which is to say, none.
- **The review produces words.** Forty minutes of discussion, genuine engagement, nothing assigned. Next month the same four items get discussed with the same energy.

> **A RAID log is not a record of what you are worried about. It is a claim that you can see things coming. The workbook grades that claim.**

That grade is one number: **the share of your open issues that were on the risk register before they happened**. Everything else in the file measures whether you are maintaining it. That measures whether it is any good, and it is the only figure worth showing anyone.

A log where nothing that went wrong was ever predicted is not a risk register. It is a diary with a governance function. There is no shame in discovering that — most of them are — but there is not much point continuing to maintain it either.

What A and D stand for is a real disagreement, not pedantry.

Most people learn RAID as Risks, Assumptions, Issues, Dependencies. RAIDLOG — the firm that has thought hardest about this in public — uses Risks, *Actions*, Issues, *Decisions*. The reasoning is worth understanding before you pick.

Their case, put fairly

Assumptions and dependencies, they argue, are identified at the start of a project, fed into the schedule and the risk plan, and then monitored. Actions and decisions have to be actively managed every single week, constantly captured and constantly tracked. A log is a tool for active management, so it should hold the things that need managing.

Assumptions and dependencies do not get dropped in their scheme. They get folded into Risks, on the grounds that both carry inherent risk: if they do not hold, the project is affected. And some project managers, they note, prefer to track dependencies in the schedule instead, which is where the scheduling consequences actually live.

That is a coherent position and it is built on a real observation. A RAID log with no actions coming out of it is a reading group.

Where we land, and why

We keep assumptions and dependencies as their own categories, and we add Actions as a fifth sheet. Not because their argument is wrong, but because of what happens in practice to anything filed under Risks.

> **Folding assumptions into risks does not raise the attention they get. It lowers it — because they now compete with things that are on fire, and they lose that competition every week.**

The two categories that rot most quietly are exactly the two being folded away. An assumption nobody has tested and a dependency nobody has confirmed are both invisible until delivery, and neither one shouts. Sitting in a risk register sorted by score, they sink. Sitting on their own sheet with their own review slot, they at least get looked at.

There is also a practical difference in what you *do* about them. A risk gets a response — accept, reduce, transfer, avoid. An assumption gets a test and a date. A dependency gets a confirmation from another human being. Those are three different actions, and merging the categories tends to merge the actions into the vaguest of the three.

On actions, though, they are simply right, so the workbook has an Actions sheet and the review is graded on whether it produced any. Decisions live in a separate log — if you are running the Operating Cadence workbook, that is the Decision Log sheet in it.

Four columns do almost all of the work. Most templates have none of them.

Every RAID template gives you a description field, an owner and a status. Those are necessary and they change nothing. These four are what make each category behave differently from the others.

Risks — the trigger date

THE DAY YOU WOULD KNOW

Not a due date. The date by which the thing has either happened or stopped being possible. "Vendor misses the integration date" has a trigger: the integration date. "The market shifts" has none, which is the tell.

Without it a risk cannot be closed, so the register only ever grows and nobody reads it by month three. With it, every risk eventually forces a decision — it became an issue, or it did not happen and you were right to watch it. Both are closures. Both are worth saying out loud, because a risk closed as "did not happen" is evidence your judgement is calibrated, and nobody ever collects that evidence.

Assumptions — how you would know it is wrong

THE TEST

Not "how confident are we" on a scale. A concrete observable: *their first sample file has more than 5% null keys*. Something a person could check on a Tuesday.

If you cannot write that sentence, what you have is a belief rather than an assumption, and beliefs cannot be tested, only disproved by delivery. This one field converts the most useless tab in most RAID logs into the most useful, and it takes about a minute per row.

Dependencies — have they confirmed

IN WRITING, VERBALLY, OR NOT AT ALL

Three states, and the difference between them is enormous. A written confirmation survives the other person being reassigned. A verbal one survives until they are. An assumed dependency — where the other party does not know they are on your critical path — is not a dependency at all. It is a hope with a due date.

The unconfirmed count is, in our experience, the single most common way a plan is wrong while everybody involved is being completely honest.

Issues — was it on the risk register

THE ONE THAT GRADES THE FILE

When you log an issue, put the risk ID it came from. If it did not come from one, write No. That is the whole mechanism, and it costs four seconds per issue.

Over a quarter it produces the only honest measure of whether your risk register is doing anything. It also changes behaviour immediately, because writing "No" repeatedly is uncomfortable in a way that no amount of process guidance manages to be.

Twenty minutes, weekly, and not in alphabetical order.

RAID is alphabetical for the same reason acronyms usually are. Reviewing in that order means spending your freshest attention on things that have not happened yet, while something is actively on fire.

Reviewing out of acronym order is RAIDLOG's idea and it is a good one — they run issues, risks, decisions, actions. Ours ends with the two categories we kept, deliberately, so the quietest things get a slot rather than the leftovers.

ORDER	WHAT YOU ASK	THE FAILURE IT CATCHES
1. Issues	What is open, what is overdue, and for each one — was it on the risk register?	Issues that quietly become conditions of the project rather than problems with it. If an issue has no fix-by date, nobody has agreed when it stops being true.
2. Risks	Whose trigger date has passed? What scores high with nothing written down?	A passed trigger is either an unlogged issue or a risk that never existed. Both need a decision that day, and both are invisible without the date.
3. Dependencies	What is due soon and only verbally agreed? What has nobody confirmed at all?	The plan being wrong while everyone is honest. Catch it three weeks out and it is an email; catch it on the day and it is a slipped delivery.
4. Assumptions	What is past its validate-by date and still untested?	Risks you decided not to look at. This slot exists because assumptions never win attention against anything urgent, so they need a place where nothing urgent is allowed.

Two rules borrowed, and one added

The two-minute rule. If something surfaced in the review can be done in under two minutes, do it in the review. Straight from *Getting Things Done* by way of RAIDLOG, and it removes perhaps a third of what would otherwise become actions.

Time-boxing. An action open past two weeks gets flagged. Not because two weeks is magic, but because an action that old is not being worked on — it is being carried, and carrying is what a review is supposed to stop.

And ours: the review must produce actions or it did not happen. The workbook counts them. A review that generated no actions two weeks running means either the project is genuinely calm, or the review has become a status meeting wearing a RAID log for cover. You will know which.

One person owns the log.

RAIDLOG put it well: it belongs to the person responsible for getting the initiative done. Not the PMO, not whoever has the spreadsheet open. A log owned by a function rather than a person gets maintained rather than used, and those look identical from the outside for about two months.

Six things people say in a RAID review.

All of them sound reasonable. That is precisely why the log rots while everybody behaves well.

“That’s more of a risk than an issue.”

USUALLY MEANS It has already happened and calling it a risk keeps it in the future tense, where nobody has to fix it this week. The category argument is doing emotional work.

WHAT YOU SAY

> "Has it happened yet?" One question, binary answer. If yes it is an issue and it needs a fix-by date. Category debates in a review are almost always avoidance with a vocabulary.

“We’re monitoring it.”

USUALLY MEANS Nobody is monitoring it. There is no date on which anybody will look, so the monitoring consists of the sentence being said again next month.

WHAT YOU SAY

> "What date would we know either way?" That is the trigger date, and asking for it converts a permanent worry into something that will eventually close.

“They’ve said they’ll do it.”

USUALLY MEANS Somebody said something encouraging in a meeting six weeks ago. It was true when they said it and may not have survived their reprioritisation since.

WHAT YOU SAY

> "In writing, or verbally?" Not a challenge to anyone's honesty — a question about durability. Verbal commitments do not survive the person who made them being moved, and people get moved.

“That’s a known unknown, we can’t plan for it.”

USUALLY MEANS

The speaker is right about uncertainty and wrong about the conclusion. You cannot plan the outcome. You can absolutely plan the date on which you find out.

WHAT YOU SAY

> "Agreed we can't control it. When do we find out?" Then log that date. Accepting a risk is a legitimate response. Accepting it without a review date is just not writing it down.

“Can we take that offline?”

USUALLY MEANS

Sometimes correct — a twenty-minute review should not absorb a working session. Sometimes it means the discussion is uncomfortable with these particular people present.

WHAT YOU SAY

> "Fine — who, and by when?" That is an action, which is what the review is for. Offline with no name and no date is where RAID items go to become permanent.

“Nothing’s changed since last week.”

USUALLY MEANS

Sometimes genuinely true. More often it means nobody has looked, and "nothing changed" is what an unexamined item looks like from the outside.

WHAT YOU SAY

> Update the last-reviewed date anyway, and mean it. The workbook flags anything unreviewed past your threshold, so a log full of untouched items becomes a number rather than a feeling — and it is usually a worse number than anyone expects.

Do not migrate your old one.

The instinct is to copy the existing RAID log across. Resist it. Most of what is in there is exactly the material that made the old one unreadable, and importing it imports the habit too.

SESSION 1

Issues and dependencies only. Ninety minutes.

These are the two you can fill in from memory, and they produce the fastest return — the unconfirmed dependency count alone usually justifies the session. Do not touch risks yet.

SESSION 2

Risks, but only ones with a trigger date.

If you cannot name the date you would know, do not log it. This will feel like you are losing risks. You are losing the ones that were never actionable, and a register of eight closeable risks beats forty permanent ones by a distance.

SESSION 3

Assumptions, with the test written first.

Write "how we would know it is wrong" before writing the assumption itself if that helps. Anything you cannot write a test for goes in a notes file, not this sheet. Expect this session to be the uncomfortable one.

WEEK 4 ON

Twenty minutes a week, in order, with actions.

Then look at The Review sheet. The first honest reading tends to be poor and that is the baseline, not the verdict. What matters is whether "issues you saw coming" moves over a quarter.

> **The register should get smaller before it gets bigger.**
>
> A good first month closes more than it opens, because most of what was in the old log was either already resolved, never real, or written in a form that could not be closed. If your count only goes up, the closure discipline has not landed yet.

Three tools, three different failures.

This is the third download in the CNNCTD toolkit, and each one catches something the other two cannot see.

TOOL	MEASURES	THE FAILURE IT CATCHES
The Operating Cadence	Coordination	The record drifting away from reality while everybody stays polite. Symptom: the board is a description of last Tuesday.
Founder Dependency Audit	Concentration	Too much of the business running through one person. Symptom: nothing moves during a fortnight's leave.
The RAID Log	Foresight	Nothing that goes wrong having been anticipated. Symptom: every problem arrives as a surprise and is handled well.

They overlap at one point worth naming. If your RAID log is stale, the cause is usually coordination rather than diligence — the review is not happening because no rhythm holds it. And if one person is the only source of every risk on the register, that is concentration, and it means the register reflects one person's blind spots exactly.

What we actually think

The RAID log has a bad reputation among people who have had one imposed on them, and mostly it is deserved. Imposed logs are compliance artifacts. A log kept by the person who has to deliver the thing, reviewed for twenty minutes a week, producing three actions, is a completely different object that happens to share a name.

Four reviews in, send the file back.

The last sheet is Snapshot. Eight numbers fill themselves in, including the one that matters — the share of your open issues that were on the risk register first. Three boxes are yours.

Email the workbook to connect@cnnctd.work. We read it before we talk, so the conversation starts from your actual log rather than a description of how you intend to run projects.

The third question — what went wrong that was not on any of these sheets — is the one we care most about. Every RAID log has a blind spot, and it is usually the same shape as the person maintaining it. Naming what got missed is worth more than the eight numbers put together.

CNNCTD, LLC · cnnctd.work · We do not share this file, we do not sell it on, and we do not add you to anything.