



PAST THE DIAGRAM

FREE · NO SIGNUP

The Architecture Pattern Pack

Seven artifacts from a full TOGAF engagement — each a blank template with the worked example beside it.

1	Stakeholder Map & Concerns Matrix	PHASE A
2	Architecture Principles as Decision Filters	PHASE A
3	Statement of Architecture Work	PHASE A
4	Business Capability Map	PHASE B
5	Data Duplication Audit	PHASE C
6	Architecture Contract	PHASE G
7	Dispensation Register	PHASE G

lerutaolo.co.za/pastthediagram

youtube.com/@pastthediagram

TOGAF® is a trademark of The Open Group. Independent educational material.



The Architecture Pattern Pack

Seven artifacts from a full TOGAF engagement, as blank templates with a worked example beside each one.

These are the actual deliverables produced across the *TOGAF 10 in Practice* series on Past The Diagram, run end to end on **Karoo Retail Group** — a fictional composite retailer built from patterns common to mid-size retail estates. The company is invented. The problems are not.

Why these seven

Most architecture templates fail for the same reason: they show you an empty grid and leave you to guess what belongs in it. A capability map with nothing in it teaches nothing. So every artifact here is paired with the filled-in Karoo version, and the reasoning for what went in and what was deliberately left out.

#	ARTIFACT	ADM PHASE	THE QUESTION IT ANSWERS
1	Stakeholder map & concerns matrix	A	Who can sink this, and what do they actually care about?
2	Architecture principles	A	How do we settle arguments without re-litigating them monthly?
3	Statement of Architecture Work	A	What is in scope, and what did we explicitly refuse?
4	Business capability map	B	What does this business <i>do</i> , independent of who does it?
5	Data & application audit	C	Where does the same fact live more than once?
6	Architecture contract	G	What do delivery and architecture owe each other?
7	Dispensation register	G	How do we say "not this time" without being routed around?

How to use them

Work in order. Each artifact assumes the one before it exists — the capability map is uninterpretable without the principles, and the architecture contract has nothing to enforce without a Statement of Architecture Work.

Two rules that matter more than the templates:

- 1. Fill the "deliberately excluded" fields.** Every template has one. A scope document that only lists what is in scope has not fenced anything, because the argument you will actually have is about something nobody wrote down.
- 2. Keep them short enough to be read.** If an artifact runs past two pages, it has become a document that exists to prove work happened rather than to change a decision. The Karoo examples are deliberately terse.

Format

The pack ships as a print-ready PDF and as plain Markdown. Use the PDF to read or print; use the Markdown to drop an artifact into Confluence, Notion, a wiki, a repo, or a Word document without fighting a layout. Edit freely — the PDF is generated from those same Markdown files, so nothing is locked away.

The episodes these came from

Each artifact names the episode that builds it, if you want the reasoning rather than just the shape.

TOGAF® is a registered trademark of The Open Group. This pack is independent educational material and is not affiliated with, endorsed by, or accredited by The Open Group.

Karoo Retail Group is a fictional composite. Any resemblance to a specific organisation is coincidental.

1

PHASE A

Stakeholder Map & Concerns Matrix

ADM Phase A — Architecture Vision · Episode 2

The purpose of this artifact is not to list everyone involved. It is to find the people who can stop the work, and to write down what they actually care about before they tell you at the worst possible moment.

A stakeholder who can veto and whose concern you never recorded is the single most common cause of an architecture that gets built and then shelved.

Template

STAKEHOLDER	ROLE / POWER	THEIR REAL CONCERN	HOW THEY'LL JUDGE SUCCESS	ENGAGEMENT
-----	Can they <i>veto</i> , <i>fund</i> , <i>block</i> , or only <i>comment</i> ?	Not their job title's concern — theirs	The measure they will personally quote	Decide / consult / inform
-----	-----	-----	-----	-----
-----	-----	-----	-----	-----

Deliberately excluded — who is *not* a stakeholder on this engagement, and why. Write this down. It is the field that prevents scope drift by committee.

...

Concerns that conflict — list the pairs that cannot both be fully satisfied, and which one the sponsor has said wins.

CONCERN A	CONCERN B	WHICH WINS, AND WHO DECIDED
-----	-----	-----

Worked example — Karoo Retail Group

STAKEHOLDER	ROLE / POWER	THEIR REAL CONCERN	HOW THEY'LL JUDGE SUCCESS	ENGAGEMENT
Group CFO	Funds it, can cancel it	That this becomes a two-year programme with no visible return	A working thing in production inside the financial year	Decide
Head of Ecommerce	Can block cutover	That her roadmap gets frozen while "architecture happens"	Her team ships in every quarter of the migration	Decide
Head of Retail Ops	Can block cutover	Tills cannot go down; a store outage is revenue and staff chaos	Zero point-of-sale downtime during migration	Decide
Loyalty Product Lead	Owns the loudest deadline	A campaign date already announced externally	The campaign ships on the announced date	Consult
Data Protection Officer	Can veto late	Customer records duplicated across systems with no single deletion path	One demonstrable path to erase a customer	Decide
Store managers	No formal power, total practical power	Anything that adds steps at the till will be worked around	Nothing changes at the till for staff	Inform
Group CTO	Sponsor	That the estate stops accumulating point-to-point integrations	New systems integrate through the shared record, not directly	Decide

Deliberately excluded

Supply chain and warehouse systems. They touch the customer record only via fulfilment status, and pulling them in doubles the estate under analysis for a read-only dependency. Revisit at the next cycle — recorded so nobody re-opens it mid-engagement as though it were an oversight.

Concerns that conflict

CONCERN A	CONCERN B	WHICH WINS, AND WHO DECIDED
Loyalty campaign date	One customer record before any new consumer	Campaign date wins once, under a time-boxed dispensation (see artifact 7) — CFO and CTO jointly
Zero till downtime	Fastest possible migration	Zero till downtime, always — Retail Ops, unchallenged

The two mistakes this artifact exists to prevent

Listing roles instead of people's actual concerns. "The CFO is concerned with cost" is a job description, not a concern. "The CFO has been burned by a programme that ran two years without shipping" is a concern, and it tells you that your roadmap needs a visible delivery early rather than a perfect one late.

Leaving the conflicts blank. Every real engagement has at least one pair of concerns that cannot both be fully met. If you do not name them and get a ruling early, you will discover them during delivery, when the cost of choosing has gone up and the person who could have chosen has moved on.

2

PHASE A

Architecture Principles as Decision Filters

ADM Phase A — Architecture Vision · Episode 2

Most principle sets are useless, and they are useless in a predictable way: they are statements nobody would ever disagree with. "We will be secure." "We will prefer reuse." "We will be cloud-first." Nothing in a real argument is settled by any of them, because the opposing position never claimed to be insecure.

A principle earns its place only if it **rules something out**. If you cannot name a decision it forbids, it is a value, not a principle — delete it.

Template

Use one block per principle. Five to eight is plenty; more and nobody recalls them under pressure, which is the only time they matter.

P1 — <short imperative name>

Statement — one sentence, in the imperative.

Rationale — the specific pain this exists to prevent. Reference the real incident or cost if there is one.

Implications — what teams must now do differently. Concrete.

This principle forbids — name at least one thing that would otherwise be reasonable. *If this is blank, the principle is not a principle.*

When it does not apply — the honest exception. A principle with no stated boundary gets applied absurdly once, then ignored permanently.

Worked example — Karoo Retail Group

P1 — One fact, one owner

Statement — Every business fact has exactly one system of record; everything else holds a copy that is explicitly marked as a copy.

Rationale — Nine systems each held a customer record and each believed it was authoritative. Reconciling them was a standing monthly cost, and a deletion request could not be honoured with confidence.

Implications — New systems read the customer from the shared record rather than storing their own. Where a local copy is unavoidable for performance, it is read-only and carries the source's version.

This principle forbids — a new service creating its own customer table because the shared record's API is slower than a local join. That is precisely the trade this principle refuses.

When it does not apply — analytical stores and warehouses, which are copies by definition and are not consumed for transactional decisions.

P2 — Buy the boring, build the differentiating

Statement — Build only what a customer would notice us doing better than a competitor; buy or adopt everything else.

Rationale — Engineering capacity was being spent on a bespoke scheduler and a homegrown feature-flag system while the checkout experience went unimproved.

Implications — Every build proposal names the customer-visible difference it creates. If it cannot, procurement is the default path.

This principle forbids — writing our own identity provider, message broker, or job scheduler, however satisfying.

When it does not apply — where no viable product exists, or where the licensing model scales worse than our own maintenance cost. Both must be shown, not asserted.

P3 — No cutover the business can feel

Statement — Every migration step must leave the estate in a working state that could be run indefinitely.

Rationale — The business is open seven days a week. There is no weekend to switch anything off, and a rollback plan that has never been exercised is a document, not a plan.

Implications — Work is sequenced as transition architectures, each shippable and each independently valuable. Strangler patterns over big-bang replacement.

This principle forbids — a migration design whose intermediate states only work if the next step lands on schedule.

When it does not apply — genuinely isolated internal tools with no store or customer dependency.

P4 — Standards have an escape hatch, and it expires

Statement — Any team may deviate from a standard through a recorded dispensation with a named owner and an expiry date.

Rationale — The previous review board had no legitimate exception path, so teams stopped bringing decisions to it and deviated silently. Governance with no escape hatch does not produce compliance; it produces invisibility.

Implications — Dispensations are cheap to request, publicly visible, and tracked to expiry. Accumulated dispensations against one standard are treated as evidence the standard is wrong.

This principle forbids — an indefinite exception, and equally forbids refusing all exceptions.

When it does not apply — regulatory and data-protection controls, which are not dispensable.

The test

Take your last three contested technical decisions. For each, ask which principle would have settled it. If the answer is "none of them", the set is decorative — rewrite it around the arguments you actually have.

3

PHASE A

Statement of Architecture Work

ADM Phase A — Architecture Vision · Episode 2

This is the scope fence. It is the artifact that lets you say "that is a new engagement" without it sounding like a refusal.

Keep it to two pages. A Statement of Architecture Work that nobody finishes reading fences nothing.

Template

Engagement name · **Sponsor** (the one person who can approve scope changes) · **Architect** · **Period**

1. Why now

Two or three sentences. The business event that made this urgent — not a general aspiration. If you cannot name what changed, the engagement has no deadline and will drift.

2. Scope — in

The domains, systems, and processes this engagement will produce architecture for. Name systems, not themes.

3. Scope — explicitly out

The things a reasonable person would assume are included, and are not. **This section is longer than section 2 in a good document.** Every line here is an argument you will not have twice.

4. Constraints

What is fixed and not up for redesign — contractual, regulatory, technical, or calendar.

5. Deliverables

One line each, with the phase that produces it.

6. What "done" looks like

Observable conditions, not documents produced.

7. Non-goals that will be proposed anyway

The predictable scope-creep requests, pre-refused with the reason.

8. Approval

Sponsor, date. Changes to scope require a new signature, not an email.

Worked example — Karoo Retail Group

Engagement Customer Record Consolidation · **Sponsor** Group CTO · **Architect** — · **Period** One financial year

1. Why now

The board has committed publicly to an omnichannel programme, and a data-protection audit found no single path to erase a customer. Both land in the same year, and both fail for the same underlying reason: the customer exists nine times.

2. Scope — in

- Point-of-sale (in-store transactions and returns)
- Ecommerce platform (accounts, orders)
- Loyalty platform (membership, points)
- The customer record itself: ownership, schema, lifecycle, and access path
- Integration between the three consumers above and the shared record

3. Scope — explicitly out

- **Supply chain and warehouse.** Read-only fulfilment status only. Not re-platformed, not re-integrated.
- **The ecommerce front end.** Its data access changes; its UX does not.
- **Payments and PCI scope.** Untouched. Any change here is a separate engagement with its own approval.
- **Marketing automation.** Consumes the record after this work; not migrated as part of it.
- **Store hardware and the till application.** No change at the till. This is a hard constraint from Retail Ops, not a preference.
- **Analytics warehouse.** Continues to receive copies. Not a system of record and not in scope for consolidation.

4. Constraints

- The estate operates seven days a week. No scheduled full outage is available.
- The loyalty campaign date is externally announced and cannot move.
- Data protection: a customer erasure request must be demonstrably satisfiable at the end of every transition state, not only at the end of the programme.

5. Deliverables

DELIVERABLE	PHASE
Stakeholder map and concerns matrix	A
Architecture principles	A
Business capability map, with the contested capabilities marked	B
Data ownership model and the duplication audit	C
Application interaction model, target state	C
Technology standards catalogue	D
Transition architectures and dependency-ordered roadmap	E / F
Architecture contracts, one per delivery team	G
Dispensation register	G

6. What "done" looks like

- One system of record for the customer, named and owned by a named person.
- Point-of-sale, ecommerce and loyalty all read that record rather than a local copy.
- A single erasure path exists and has been executed end to end at least once.
- No point-of-sale downtime attributable to the migration.
- Every remaining local copy is documented, read-only, and versioned to source.

7. Non-goals that will be proposed anyway

- *"While we're in there, let's replace the loyalty platform."* No. Replacing a consumer and consolidating the record at once means neither can be rolled back independently.
- *"Can we add a single customer view for marketing?"* Not in this engagement. It becomes trivial afterwards and expensive during.
- *"Should we move to microservices?"* Out of scope as an objective. Service boundaries change only where consolidation requires it.

8. Approval

Sponsor: _____ Date: _____

Why section 3 is the important one

Everyone writes section 2. The engagements that sprawl are the ones where section 3 was left thin, because scope creep almost never arrives as a request for something obviously outside the work. It arrives as something adjacent and apparently free — *"we're already touching loyalty, so..."* — and without a pre-written refusal and a reason, the honest answer in the room is "I suppose so".

4

PHASE B

Business Capability Map

ADM Phase B — Business Architecture · Episode 3

A capability is something the business *does*, stated so that it stays true if you replaced every system and reorganised every team tomorrow. "Accept payment" is a capability. "Stripe integration" is not. "Warehouse team" is not.

The map earns its keep in exactly one way: it lets you argue about **where a capability lives** without the argument becoming about which department owns which system.

Template

Two levels is usually enough. Three is a research project.

```
<Value stream>
├─ L1 Capability
│   └─ L2 ...
│       └─ L2 ...
└─ L1 Capability
```

Then rate only what you are going to act on:

CAPABILITY	MATURITY (1–5)	BUSINESS CRITICALITY	DUPLICATED IN	VERDICT
-----	-----	High / Med / Low	which systems perform it today	Invest / hold / consolidate / retire

Contested capabilities — the ones performed in more than one place, where nobody agrees who owns them. These are the whole point of the exercise; list them explicitly rather than burying them in the table.

Worked example — Karoo Retail Group

```
Sell to a customer
├─ Identify the customer          <- contested
├─ Present products and price
├─ Accept payment
└─ Fulfil the order

Keep a customer
├─ Record customer consent      <- contested
├─ Accrue and redeem loyalty value
├─ Handle returns and refunds
└─ Communicate with the customer <- contested

Run the estate
├─ Manage stock position
├─ Report on trading
└─ Erase a customer on request   <- contested, and performed nowhere completely
```

CAPABILITY	MATURITY	CRITICALITY	DUPLICATED IN	VERDICT
Identify the customer	2	High	POS, ecommerce, loyalty (3 independent implementations)	Consolidate

CAPABILITY	MATURITY	CRITICALITY	DUPLICATED IN	VERDICT
Record customer consent	1	High	ecommerce, loyalty, marketing — each with different wording	Consolidate
Communicate with the customer	2	Medium	loyalty, marketing, ecommerce transactional mail	Consolidate later — depends on the record
Erase a customer on request	1	High	performed partially in 3 systems, completely in none	Invest
Accept payment	4	High	POS, ecommerce	Hold — duplication here is deliberate and works
Accrue and redeem loyalty value	3	Medium	loyalty only	Hold
Manage stock position	3	High	warehouse	Out of scope

Contested capabilities

Identify the customer. Performed three times, three different ways. In store by loyalty card scan, online by account login, in the loyalty app by phone number. None of them can tell you they are looking at the same person. This is the capability the entire engagement exists to consolidate.

Erase a customer on request. Each system can delete its own rows. No system can assert the person is gone from the estate. A capability the business is legally required to have and does not actually have is the strongest possible argument to a funding committee — stronger than any efficiency case.

The two failure modes

Mapping the org chart. If your L1 capabilities correspond one-to-one with departments, you have drawn the organisation, not the business. The test: if the company reorganised next month, would the map need redrawing? It should not.

Mapping everything. A complete capability map of a mid-size retailer runs to hundreds of boxes and changes nothing, because no one can hold it in mind. Map the value streams the engagement touches, rate only what you intend to act on, and leave the rest unrated rather than guessing.

The output that matters is the contested list. Everything else is context for it.

5

PHASE C

Data Duplication Audit

ADM Phase C — Data & Application Architecture · Episode 4

The artifact that turns "our data is a mess" into a fundable, arguable, finishable piece of work.

You are looking for one thing: **the same business fact stored in more than one place, with no agreement about which one is right.** Not tables. Not databases. Facts.

Template

Pick one entity at a time. Customer first, almost always — it is the entity with the most duplication and the only one with a legal deletion requirement.

Entity: <name>

#	SYSTEM	WHAT IT STORES	CREATED WHEN	UPDATED BY	CLAIMS TO BE AUTHORITATIVE?	CAN IT BE DELETED?
1	-----	-----	-----	-----	-----	-----

Reconciliation cost today — what is spent keeping these in agreement: people, jobs, and the failure that happens when they disagree.

The disagreement test — when two of these hold different values, which one wins today? If the answer is "whichever was updated last" or "nobody knows", write that down. It is the finding.

Target — one system of record, named. Every other row becomes one of: *read-through* (no local storage), *cached copy* (read-only, versioned to source), or *retired*.

#	SYSTEM	TARGET ROLE	MIGRATION STEP
---	--------	-------------	----------------

Worked example — Karoo Retail Group

Entity: Customer

#	SYSTEM	WHAT IT STORES	CREATED WHEN	UPDATED BY	AUTHORITATIVE?	DELETABLE?
1	POS	name, loyalty card no.	first in-store scan	store staff at till	claims yes	no — needed for returns

#	SYSTEM	WHAT IT STORES	CREATED WHEN	UPDATED BY	AUTHORITATIVE?	DELETABLE?
2	Ecommerce	email, name, address, password	online registration	customer self-service	claims yes	partially
3	Loyalty platform	phone, name, points balance	app signup	customer + campaigns	claims yes	no — points are a liability
4	Marketing automation	email, consent flags	first campaign import	import job	no	yes
5	Returns tool	name, order ref, contact	first return	service desk	no	no
6	Call-centre CRM	everything, typed by hand	first call	agents	claims yes	yes
7	Ecommerce guest orders	email + address only	guest checkout	nobody	no	orphaned
8	Analytics warehouse	copies of 1–7	nightly	ETL	no	rebuilt nightly
9	Email service provider	email, engagement	sync	sync job	no	yes

Reconciliation cost today A monthly manual exercise to align loyalty balances against POS transactions, plus a standing service-desk workload for customers who exist twice and cannot see their own points. When 1 and 3 disagree, the customer is told the balance from whichever system the agent happened to open.

The disagreement test There is no rule. Precedence is decided by which system the person answering looked at. That single sentence justified the engagement more effectively than any cost model.

Target

#	SYSTEM	TARGET ROLE	MIGRATION STEP
—	New: shared customer record	system of record	Transition 1
1	POS	read-through, card no. as lookup key	Transition 2
2	Ecommerce	read-through	Transition 3
3	Loyalty	read-through; points stay local (they are its own fact)	Transition 4
4	Marketing	consumer of the record; consent moves to the record	after Transition 4
5	Returns	read-through	Transition 3
6	Call-centre CRM	read-through; hand-typed customer creation removed	Transition 4
7	Guest orders	linked to the record on first identification, else expired	Transition 3

#	SYSTEM	TARGET ROLE	MIGRATION STEP
8	Warehouse	unchanged — a copy by design, not a system of record	—
9	ESP	unchanged, receives from the record	after Transition 4

Note that **points stay in the loyalty platform**. A points balance is genuinely that system's own fact, not a duplicated customer attribute. Consolidating it into the record because it sits next to customer data would be the same mistake in the opposite direction.

How to run it without a six-month discovery

Do not model the estate. Ask one question of each system owner: *"If this disagreed with another system about a customer's name, which one is right?"*

You will get three kinds of answer — a confident "us", a confident "them", and a pause. Every pause is a finding, and the pauses are usually the majority. The audit is mostly a record of pauses.

6

PHASE G

Architecture Contract

ADM Phase G — Implementation Governance · Episode 7

The artifact almost every organisation gets half-right.

An architecture contract is **bidirectional**. The delivery team commits to something, and architecture commits back. Written as a one-way list of obligations on delivery, it is not a contract — it is a compliance checklist, and it produces the behaviour compliance checklists always produce: teams route around it and stop asking.

The second column is the one that makes governance work, and it is the one that is almost always missing.

Template

Delivery team · Work package · Period · Architect · Signed

The delivery team commits to

#	COMMITMENT	HOW IT'S CHECKED	WHEN
1	-----	-----	-----

Architecture commits to

#	COMMITMENT	MEASURED BY	IF WE MISS IT
1	-----	-----	and this column matters — the remedy when architecture is the blocker

Review points

POINT	TRIGGER	WHAT IS REVIEWED	WHO ATTENDS
-----	not "every sprint" — a real event	-----	-----

Escalation

Who decides when the two sides disagree, and how long that decision may take.

Worked example — Karoo Retail Group, Loyalty delivery team

Work package Loyalty reads the shared customer record · **Period** Transition 4 · **Architect** — · **Signed** —

The delivery team commits to

#	COMMITMENT	HOW IT'S CHECKED	WHEN
1	Read customer identity from the shared record; no new local customer table	Schema review	Before build
2	Publish a <code>points.accrued</code> event rather than letting others read its database	Integration review	Mid-build
3	Keep the points balance local — it is loyalty's own fact	Schema review	Before build
4	Use the Phase D standard database platform	Automated check	Continuous
5	Raise a dispensation rather than deviating silently	Nothing to check — it is the escape hatch	Any time

Architecture commits to

#	COMMITMENT	MEASURED BY	IF WE MISS IT
1	A decision on any raised question within 5 working days	Date raised to date answered	The team proceeds with their proposed approach, and it is deemed approved
2	The reference architecture is current, not aspirational	Last-updated date visible on the document	The team may treat the stale section as non-binding
3	An architect attends the team's design session in person, not by document	Attendance	The review point is waived, not deferred
4	Give the shared record's API a documented latency expectation before build starts	Published before kickoff	Team may cache locally under a dispensation without further approval

Note what column 4 does. Every architecture commitment has a **stated consequence for architecture failing to meet it**, and the consequence always favours delivery continuing. That is what makes the contract real rather than decorative. A governance process whose only failure mode is "the team waits" is a governance process that will be bypassed.

Review points

POINT	TRIGGER	WHAT IS REVIEWED	WHO ATTENDS
Design intent	Before the first line of build	Data ownership, integration shape, standards fit	Architect + tech lead
Integration	First event published end to end	Contract of the event, versioning, consumer impact	Architect + tech lead + one consumer
Pre-release	Before store rollout	Rollback path, erasure path, till impact	Architect + Retail Ops

Three review points across a work package. Not every sprint, not every pull request — reviews attached to events where the answer can still change something.

Escalation

Unresolved after one working week: Group CTO decides, within two working days. Silence past that is treated as approval of the delivery team's position.

The test for whether yours is real

Cover the "delivery team commits to" column and read only what architecture committed. If it reads as an empty courtesy — "architecture will provide guidance" — you have a checklist. Real commitments have dates, numbers, and a named consequence when architecture is the one who misses.

7

PHASE G

Dispensation Register

ADM Phase G — Implementation Governance · Episode 7

A dispensation is a recorded, scoped, **expiring** permission to deviate from a standard.

Governance without a legitimate escape hatch does not produce compliance. It produces invisibility: teams stop asking, deviate quietly, and the architecture function finds out during an incident. The register exists so that deviation is *cheaper to declare than to hide*.

The expiry is not paperwork. A permanent exception is not a dispensation — it is an undocumented change to the standard.

Template

ID	STANDARD DEVIATED FROM	REQUESTED BY	REASON	SCOPE — EXACTLY WHAT IS PERMITTED	GRANTED	EXPIRES	OWNER	REPAYMENT
D-001	-----	-----	-----	narrow enough that it cannot be reused elsewhere	date	date, not "TBC"	a person	what must be true at expiry

Standing rules

1. **An expiry date is mandatory.** No open-ended dispensations. If it genuinely cannot expire, that is a standards change, and it goes through standards change — not here.
2. **Scope is written narrowly.** "Team X may cache customer data" is too broad. "The loyalty campaign service may hold a read-only customer name and card number for the duration of the November campaign" is a dispensation.
3. **The register is public.** Hidden exceptions teach everyone that exceptions are shameful, which is exactly how you lose visibility.
4. **Count them.** Three or more dispensations against the same standard is evidence the standard is wrong, not that three teams are difficult. Review the standard.

Expiry review

ID	EXPIRY	OUTCOME	NOTES
-----	-----	<i>Repaid / Extended once, with reason / Escalated</i>	An extension is a decision, not an administrative default

Worked example — Karoo Retail Group

ID	STANDARD	REQUESTED BY	REASON	SCOPE	GRANTED	EXPIRES	OWNER	REPAYMENT
D-001	One fact, one owner (P1)	Loyalty team	The shared record will not be live before the externally announced campaign date	The campaign service may hold a read-only copy of customer name and loyalty card number, for the campaign period only. No writes. No other attributes.	12 Feb	31 Mar	Loyalty Product Lead	Copy deleted; service reads the shared record
D-002	Standard database platform (Phase D)	Returns team	Existing returns tool is a vendor product; platform is not ours to choose	Returns tool remains on its vendor-managed store	3 Mar	Next contract renewal	Head of Retail Ops	Re-evaluated at renewal; not automatically renewed
D-003	No cutover the business can feel (P3)	POS team	One 20-minute till restart is unavoidable for the driver upgrade	Single restart, one store, outside trading hours, with rollback rehearsed	8 Apr	Single use	POS tech lead	Closed on completion

Expiry review

ID	EXPIRY	OUTCOME	NOTES
D-001	31 Mar	Repaid	Copy removed 28 Mar; loyalty now reads the record
D-003	single use	Repaid	Executed 11 Apr, no customer impact
D-002	contract renewal	Extended once	Renewal moved by 6 months; extension recorded with the new date rather than left open

What the register tells you that nothing else does

D-001 is the interesting one. A team hit a genuine standard against a genuine external deadline. Without a dispensation path there were only two outcomes: the campaign misses its announced date, or the team quietly builds a customer table and nobody hears about it for a year.

The dispensation produced a third: the deviation happened, it was narrow, it was visible, it had a date, and it was actually removed. That is what a working governance process looks like — not the absence of deviation, but deviation you can see and unwind.

If your register is empty, the most likely explanation is not that everyone complies. It is that nobody is asking.