A RESOURCE FOR DESIGN SYSTEM LEADS AND THEIR SPONSORS

Decision rights template.

Governance that speeds a system up.

One decider. One deadline. No meeting unless the door is one-way.

26	12	13
rules, each with its check	decision types in the example matrix, none needing a meeting	errors caught in the violations example

Most governance is a queue with no owner.

Teams dislike design system governance because they have met it as enforcement: a council that meets every other Thursday, a pull request waiting for four approvals. The essay behind this guide argues the opposite. Governance is decision rights, a change format and release discipline, and done well it is a throughput tool. Agents raise the stakes: a system they build with changes faster, and every change needs someone who can say yes or no.

The models this guide borrows from agree on the core. RAPID names one person who decides [S01], DACI one approver [S02], and the advice process lets the person closest to the work decide after seeking advice [S04]. Bezos adds the sorting rule: reversible decisions should be made quickly [S05].

The template turns that into a file. `decision-rights.json` gives each type of decision one decider, a deadline in business days, a record and the paths it owns. The kit checks it, generates CODEOWNERS from it and measures time to decision.

Version 1.0 / Sources checked 24 September 2026
Field Guide 06 of the Design × AI series. The matrix governs the same Acme Button, tokens and docs as Field Guides 02 to 05. Kit verified on Node 22.22 against GitHub's CODEOWNERS documentation as of 24 September 2026.

Practical guidance, not a standard. People, handles and the decision log in the kit are fictional; time-to-decision figures come from that sample log, not a real team. Prepared with AI assistance and edited by hand.

START HERE

One matrix, read by people and by GitHub.

The field names map onto the frameworks your organisation may already use. Pick the vocabulary your sponsor knows; keep the rules.

This template	RAPID (Bain)	DACI (Atlassian)	RACI	Advice process
`decider`, one role	Decide	Approver	Accountable	The person taking the decision
`advisors`	Input	Contributors	Consulted	Everyone affected, and experts
`agree`, at most one	Agree	none	none	none
`informed`	none	Informed	Informed	none
`author` (the proposer)	Recommend	Driver	Responsible	Anyone may decide

Sources: S01, S02, S03, S04. `author` as decider is this template's advice process, allowed on two-way doors only.

Writing the first matrix

Copy `decision-rights.json`, list the ten decisions your team made most often last quarter, and give each one decider. Section 01 first.

Unblocking a stalled system

Section 02 and 03: put an SLA on every decision and take meetings off every two-way door. Run the SLA report on last quarter's pull requests.

Opening the system to contributors

Section 03, rule G15. Small contributions stay fast; only new components and removals take the long route.

Enforcing it

Section 04. Generate CODEOWNERS from the matrix and turn on required code-owner review, so the decider is the reviewer GitHub asks.

Label	Meaning
`MATRIX`	A field in `decision-rights.json`.
`CHECK`	Enforced by `check-decision-rights.mjs` (error or warning).
`CODEOWNERS`	Enforced by the generated CODEOWNERS file and GitHub's code-owner review.
`SLA`	Measured by `sla-report.mjs` over the decision log.
`PRACTICE`	A working method with a review signal rather than a hard gate.

SECTION 01

Name one decider.

One decision, one person who makes it. Everyone else has a voice, and the matrix says whose.

Suggested owners: Design-system lead + sponsor

- [ ] **Every decision type names exactly one decider**

  G01 MATRIX CHECK

  "Engineering and design decide together" means nobody decides. One role in `decider`, never a list; the rest go in `advisors`.

  Evidence: Checker error `no-single-decider` on a missing decider or an array.

  RAPID: ideally one person per decision [S01]. DACI: one approver [S02]. RACI: one accountable, in some theories [S03].

- [ ] **A committee advises; it never decides**

  G02 CHECK

  A council, board or guild can advise or be informed. It cannot decide or be the escalation: a group has no one who owns the deadline.

  Evidence: Checker error `committee-decider` when the decider or escalation is a role with `members` or a group title.

  Bain allows a group D only with a protocol agreed upfront; this template is stricter. S01.

- [ ] **The decider is a named person with a GitHub handle**

  G03 MATRIX CHECK

  A role is a promise; a person keeps it. Name who holds the role today and their GitHub handle.

  Evidence: Checker error `no-person` when `person` is empty or `github` is not a valid `@handle`.

  Needed for CODEOWNERS generation. S13.

- [ ] **Every decider has a delegate**

  G04 MATRIX CHECK

  Holidays are when SLAs break. Name who decides when the decider is away or the deadline is close.

  Evidence: Checker warning `no-delegate`; the SLA report accepts the decider or the delegate.

  Recommended practice.

- [ ] **At most one veto per decision**

  G05 MATRIX CHECK

  An `agree` role must sign off, which makes it a veto. Keep it for one non-negotiable constraint, such as accessibility on a new component.

  Evidence: Checker warning `many-agree` above one; the schema caps `agree` at one item.

  RAPID: Agree is assigned sparingly. S01.

If two people can say yes, you will wait for both of them.

SECTION 02

Put a clock on every decision.

Throughput is time to decision. An SLA turns "we are discussing it" into a date, and escalation says what happens when the date passes.

Suggested owners: Design-system lead

- [ ] **Every decision type has an SLA in business days**

  G06 MATRIX CHECK

  Count from the day the request is complete to the day the decider says yes, no or not yet with a reason. Business days, so a Friday request is not late on Monday.

  Evidence: Checker error `no-sla` when `slaBusinessDays` is missing or below 1.

  Recommended practice.

- [ ] **SLAs have a cap per door**

  G07 CHECK

  The example caps two-way doors at 3 business days and one-way doors at 10. Tighten them once the report shows you meet them.

  Evidence: Checker error `sla-over-cap` against `policy.maxSlaBusinessDays`.

  Precedents: Apache votes run at least 72 hours [S06]; a Rust RFC's final comment period lasts ten calendar days [S07]. The caps are defaults, not findings.

- [ ] **Advice is sought and recorded, not obeyed**

  G08 MATRIX PRACTICE

  Advisors are asked before the decision and their advice goes into the record. The decider may decide against it, and says why.

  Evidence: The ADR template's Advice received table, one row per advisor in the matrix.

  The advice process: seek advice from those affected and from experts; it is not binding. S04.

- [ ] **Silence has a meaning**

  G09 MATRIX PRACTICE

  Advisors get an advice window. When it closes without an answer, the record says "no response" and the decider proceeds. Waiting for everyone is how a two-day decision takes a month.

  Evidence: `adviceWindowBusinessDays` on advice-process decisions; "no response by" rows in ADRs.

  Apache lazy consensus: silence gives assent. S06.

- [ ] **Escalation goes to one other person**

  G10 CHECK

  When the decider misses the SLA or a disagreement will not settle, one named role above them decides. Never the decider, never a group.

  Evidence: Checker error `no-escalation` when missing or equal to the decider; `committee-decider` when it is a group.

  Recommended practice.

A decision without a date is a queue.

SECTION 03

Size the door before you open it.

Most design system decisions are reversible. Save the slow process for the few that are not.

Suggested owners: Design-system lead + engineering lead

- [ ] **Every decision type is a one-way or a two-way door**
  G11 MATRIX CHECK
  One-way: hard to reverse once consumers depend on it. Two-way: a later release can undo it.
  Evidence: Checker error `no-door` unless `door` is `one-way` or `two-way`.
  Bezos, Type 1 and Type 2 decisions. S05.

- [ ] **One-way doors get a written record**
  G12 CHECK
  An ADR for removals, policy changes and waivers; an RFC for a new component.
  Evidence: Checker error `one-way-needs-record` on `record: "pr"`. Templates in `templates/`.
  Nygard's ADR sections and statuses [S08]. Curtis uses ADRs to evolve component contracts [S09].

- [ ] **Two-way doors are decided without a meeting**
  G13 CHECK
  The pull request is the meeting: the decider reads, asks in the thread and decides inside the SLA.
  Evidence: Checker error `two-way-meeting` when a two-way door has `meeting: true`.
  Bezos: reversible decisions made quickly by individuals or small groups. S05.

- [ ] **The author decides small, reversible changes after seeking advice**
  G14 MATRIX CHECK
  Docs fixes need no decider but their author, who asks the docs owner and merges within a day.
  Evidence: Checker error `author-misuse` unless the door is two-way, with advisors and a short advice window.
  The advice process: anyone can decide, after seeking advice. S04.

- [ ] **Small contributions stay fast; new components take the long road**
  G15 PRACTICE
  Fix, small enhancement, large enhancement, new component: only the last goes through proposal criteria.
  Evidence: D01 and D02: 1 and 2 business days, PR only. D06: an RFC, 10 business days.
  Curtis's contribution sizes [S10]; GOV.UK proposal criteria [S11]; Frost's governance process [S12].

- [ ] **No RFC for a reversible change**
  G16 CHECK
  Run the one-way process on two-way decisions and every prop becomes a proposal.
  Evidence: Checker warning `heavy-two-way` on a two-way door with `record: "rfc"`.
  Bezos: heavy-weight process on Type 2 decisions leads to slowness. S05.

Save the slow process for the doors you cannot walk back through.

SECTION 04

Wire the matrix into the repository.

A decision-rights page on a wiki is advice. CODEOWNERS makes GitHub ask the decider on every pull request that touches their paths.

Suggested owners: Engineering lead

- [ ] **CODEOWNERS is generated from the matrix**

  G17 CODEOWNERS

  Each decision's `paths` become CODEOWNERS lines with the decider and the delegate. Require code-owner review in branch protection. Any one owner on a line can approve, so agree when the delegate acts.

  Evidence: `generate-codeowners.mjs decision-rights.json --check .github/CODEOWNERS` fails in CI when the file drifts.

  GitHub: the last matching pattern wins; one approval from any listed owner satisfies required review. S13.

- [ ] **One path, one decider**

  G18 CHECK

  Two decisions may share a path only if they share a decider. Otherwise CODEOWNERS would silently pick whichever line comes last.

  Evidence: Checker error `path-conflict`; the generator refuses a matrix with errors.

  GitHub: the last matching pattern takes precedence. S13.

- [ ] **One-way doors are routed by the file they cannot avoid**

  G19 CODEOWNERS MATRIX

  CODEOWNERS cannot tell a fix from a removal in one folder. Route one-way decisions through files only they change: `src/index.ts` for new exports, `token-renames.json` and `migrations/` for removals.

  Evidence: D06, D07 and D09 in the example matrix own exactly those paths.

  Field Guide 03, rule R24: every removal appears in the migration map.

- [ ] **Author-decided paths have no required owner**

  G20 CODEOWNERS

  A pattern with no owners removes broader owners for that path, so docs fixes need only the normal review.

  Evidence: The generated file lists `/docs/` with no owner and `/docs/deprecation-policy.md` after it with the design system lead.

  GitHub: without an owner, anyone with write access can approve. S13.

- [ ] **Only patterns GitHub supports**

  G21 CHECK

  CODEOWNERS follows most gitignore rules but not negation with `!`, character ranges with `[ ]` or escaping with a backslash. Invalid lines are skipped silently.

  Evidence: Checker error `bad-pattern`.

  GitHub CODEOWNERS syntax exceptions. S13.

If GitHub does not know who decides, nobody else will remember.

SECTION 05

Run it, and measure time to decision.

The matrix is a hypothesis about how fast your system can move. Pull requests carry it; the decision log tests it.

Suggested owners: Design-system lead + release owner

- [ ] **Every pull request names its decision**

  G22 PRACTICE

  The template asks for the decision type, the decider and the due date, and links the ADR for one-way doors. Reviewers, and agents opening pull requests, classify the change before anyone reviews it.

  Evidence: `templates/pull_request_template.md` copied to `.github/`.

  Recommended practice.

- [ ] **The matrix governs itself**

  G23 MATRIX CODEOWNERS

  Changing who decides is a one-way door with its own decider, usually the sponsor. The matrix file and CODEOWNERS are routed to that person.

  Evidence: D10 in the example matrix owns `/decision-rights.json` and `/.github/CODEOWNERS`.

  Recommended practice.

- [ ] **Log every decision with opened and decided dates**

  G24 SLA

  One line per decision: id, decision type, title, opened, decided, decided by. Pull request timestamps and ADR dates are enough; the log can be generated from them.

  Evidence: `decision-log.json` in the shape of the kit's example.

  Recommended practice.

- [ ] **Report breaches and overdue decisions every week**

  G25 SLA

  Median and slowest time per decision type, breaches against the SLA and open decisions past their due date. Overdue items go to the escalation role, not to a meeting.

  Evidence: `sla-report.mjs` exits 1 when anything is overdue or decided by the wrong person.

  Kit script.

- [ ] **Decisions by the wrong person are defects in the matrix**

  G26 SLA PRACTICE

  When the council decides what the engineering lead owns, either the matrix is wrong or the team is not using it. Fix whichever it is at the next quarterly review.

  Evidence: The report's WRONG DECIDER lines; a quarterly ADR under D10 when the matrix changes.

  Recommended practice.

Governance is working when decisions are boring and fast.

APPENDIX A

The Acme matrix.

Twelve decision types from `decision-rights.json`. The checker reports 0 errors and 0 warnings on it. People and handles are fictional.

ID	Decision	Door	Decider	SLA	Record	Owns
D01	Fix a defect with no API change	two-way	eng-lead	1	PR	`/src/components/`
D02	Add an optional prop or enum value	two-way	eng-lead	2	PR	`/src/components/`, `/contracts/`
D03	Change a semantic token value	two-way	token-owner	2	PR	`/tokens/semantic/`
D04	Add or deprecate a semantic token	two-way	token-owner	2	PR	`/tokens/semantic/`
D05	Correct or extend documentation	two-way	author	2	PR	`/docs/`
D06	Add a component to the public API	one-way	ds-lead	10	RFC	`/src/index.ts`
D07	Remove or rename anything public	one-way	ds-lead	5	ADR	`/token-renames.json`, `/migrations/`
D08	Deprecate a component or prop	two-way	eng-lead	3	ADR	none
D09	Change the deprecation policy or cadence	one-way	ds-lead	10	ADR	`/docs/deprecation-policy.md`
D10	Change these decision rights	one-way	sponsor	10	ADR	`/decision-rights.json`, `/.github/CODEOWNERS`
D11	Cut a release	two-way	release-owner	1	PR	`/CHANGELOG.md`
D12	Waive an accessibility requirement	one-way	a11y-lead	3	ADR	none

SLA in business days. D06 also names the accessibility lead as its one agree role. Deprecation is a two-way door; removal is not.

APPENDIX B

What the checker catches, and what GitHub gets.

`examples/decision-rights.violations.json` is a first draft of the same matrix with the usual mistakes. The checker exits 1 with 13 errors and 4 warnings.

terminal

```
$ node scripts/check-decision-rights.mjs examples/decision-rights.violations.json
decision-rights.violations.json: 4 decisions, 13 errors, 4 warnings
  error   D01 no-single-decider: 2 deciders listed; name one
  error   D02 committee-decider: decider "ds-council" (Design system council)
          is a group; name the one person who decides
  error   D02 sla-over-cap: 10 business days is over the two-way cap of 3
  error   D02 two-way-meeting: two-way doors are decided without a meeting
  error   D03 bad-pattern: "!/tokens/semantic/dark.tokens.json" uses syntax
          CODEOWNERS does not support (!, [ ], \ or leading #)
  error   D07 one-way-needs-record: a one-way door needs an ADR or RFC, not
          only a pull request
  error   D07 path-conflict: "/tokens/semantic/**/*.json" is also claimed
          by D03 (token-owner); one path, one decider
  ... 6 more errors
  warning D02 heavy-two-way: an RFC on a two-way door slows a reversible
          decision
  ... 3 more warnings
```

Output from the kit, wrapped for print and shortened. The full run is in the kit's tests.

examples/CODEOWNERS (generated, excerpt)

```
# Generated from decision-rights.json (Acme Design System, matrix 1.0.0)
# by generate-codeowners.mjs. Do not edit by hand.
# GitHub uses the last matching pattern, so broader patterns come first.

# D05 two-way, 2bd. The author decides after seeking advice from:
# Docs owner. No required owner.
/docs/

# D01 two-way, 1bd; D02 two-way, 2bd. Decider: Design system
# engineering lead (Mikko Laine), delegate Aino Virtanen.
/src/components/ @mikko-laine @aino-virtanen

# D09 one-way, 10bd. Decider: Design system lead (Aino Virtanen),
# delegate Mikko Laine.
/docs/deprecation-policy.md @aino-virtanen @mikko-laine
```

3 of 11 patterns; comment lines wrapped and the header shortened for print. Pass `--no-delegates` to list deciders only.

APPENDIX C

Time to decision, from the log.

The kit's sample log holds 17 fictional decisions from July to September 2026. It is there to show the report, not to benchmark anything.

terminal

```
$ node scripts/sla-report.mjs decision-rights.json examples/decision-log.json --as-of 2026-09-
24
Decisions as of 2026-09-24: 17 logged, 15 decided, 14 within SLA.
Median time to decision on two-way doors: 1 business day.

Type Door     SLA  Done  Open  Median  Slowest  Breaches
D01  two-way  1    2     0     0.5     1        0
D02  two-way  2    3     0     2       4        1
D03  two-way  2    1     1     1       1        0
D06  one-way  10   1     1     9       9        0
D07  one-way  5    1     0     4       4        0
D08  two-way  3    1     0     3       3        0
OVERDUE DR-2026-057 (D03) "Raise color.text.muted contrast": due 2026-09-23,
  1 business day(s) late
OVERDUE DR-2026-056 (D06) "Add Toast": due 2026-09-15, 7 business day(s) late
WRONG DECIDER DR-2026-050 (D08) decided by ds-council; the matrix names eng-lead
```

6 of 10 rows shown. The report exits 1: two decisions are overdue and one was made by a group the matrix does not name.

templates/adr.md (headings)

```
# ADR-NNNN: <short noun phrase>
- Decision type / Door / Status / Decider / Opened / Due / Decided
## Context
## Options considered
## Advice received      (one row per advisor in the matrix)
## Decision             ("We will ...", active voice)
## Consequences
## Follow-up            (contract, migration map, changelog, log entry)
```

Nygard's five parts, plus the advice table the advice process needs and the dates the SLA report reads.

KEEP WITH THE MATRIX

Leave a quarterly review record.

One record per quarterly review of the matrix. It is the trail that shows why a decision moved, or why an SLA changed.

Matrix version / review date	**Reviewed by (sponsor, design-system lead)**
Checker run: errors, warnings (link)	**CODEOWNERS regenerated and checked (link)**
Decisions logged / decided / within SLA	**Median time to decision, two-way doors**
Overdue and escalated decisions	**Decisions made by the wrong person**
Changes to deciders, SLAs or paths (ADR link)	**Next review date**

> **Change the matrix, not the exception.**
> If the same decision keeps going to someone the matrix does not name, record the change under D10 and move the decision. A matrix people route around is worse than none.

SOURCES / MAINTENANCE

Keep the guide current.

Checked 24 September 2026. SLA precedents are open-source governance, not design system benchmarks.

S01 / Bain & Company, RAPID: bringing clarity to decision accountability
One Decide role; Agree assigned sparingly.
https://www.bain.com/insights/rapid-tool-to-clarify-decision-accountability/

S02 / Atlassian Team Playbook, DACI
One approver: "yes: one!"
https://www.atlassian.com/team-playbook/plays/daci

S03 / Wikipedia, Responsibility assignment matrix
RACI roles; one accountable per task in some theories.
https://en.wikipedia.org/wiki/Responsibility_assignment_matrix

S04 / Andrew Harmel-Law, Scaling the Practice of Architecture, Conversationally
The advice process; ADRs record the advice.
https://martinfowler.com/articles/scaling-architecture-conversationally.html

S05 / Jeff Bezos, 2015 letter to shareholders
Type 1 and Type 2 decisions; heavy process slows Type 2.
https://s2.q4cdn.com/299287126/files/doc_financials/annual/2015-Letter-to-Shareholders.PDF

S06 / Apache Software Foundation, voting
Lazy consensus; votes run at least 72 hours.
https://www.apache.org/foundation/voting.html

S07 / Rust RFCs, README
Final comment period of ten calendar days.
https://github.com/rust-lang/rfcs/blob/master/README.md

S08 / Michael Nygard, Documenting Architecture Decisions
ADR sections and statuses; superseded records are kept.
https://www.cognitect.com/blog/2011/11/15/documenting-architecture-decisions

S09 / Nathan Curtis, Component Contracts and Schemas
ADRs evolve component specs and schema versions.
https://nathanacurtis.substack.com/p/component-contracts-and-schemas

S10 / Nathan Curtis, Defining Design System Contributions
Four contribution sizes; small ones fast and autonomous.
https://nathanacurtis.substack.com/p/defining-design-system-contributions-eb48e00e8898

S11 / GOV.UK Design System, contribution criteria
Proposals useful and unique; published work usable, consistent, versatile.
https://design-system.service.gov.uk/community/contribution-criteria/

S12 / Brad Frost, A design system governance process
Default to existing components; snowflake or system.
https://bradfrost.com/blog/post/a-design-system-governance-process/

S13 / GitHub Docs, About code owners
Pattern precedence, owner approval, unsupported syntax.
https://docs.github.com/en/repositories/managing-your-repository-settings-and-features/customizing-your-repository/about-code-owners

Maintenance: recheck GitHub's CODEOWNERS rules when branch protection or rulesets change. Review the matrix each quarter with the SLA report.