Dweve
Weaving Intelligence

FOUNDATION AND DECISION GUIDE · F13

Open foundations, portability, and exit

What can be inspected, built, pinned, verified, replaced, retained, and operated when a commercial relationship ends.

DECISION SCOPE EVIDENCE OWNERSHIP PROOF EXIT

A complete guide to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

F13

CONTENTS

From the first question to a defensible decision

This 38-page guide turns 14 owned topics into an explanation, evidence requirement, proof challenge, and closing decision record.

PRIMARY READERS	READING DEPTH	DECISION THIS SUPPORTS
Open-source, architecture and continuity teams	Technical	Determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

TARGET LENGTH

9,000–13,000 words.. Page count varies with the number and depth of topics rather than forcing every subject into the same shell.

SOURCE DISCIPLINE

Dweve-specific facts are taken directly from the English website text audited 2026-07-20; web-navigation wording is humanised for print. Evaluation guidance is editorial, and scenarios and derived visuals are illustrative. Unsupported synthesis is replaced by canonical copy or omitted.

READ THIS FIRST · READER DECISION

Who should use this guide, and what decision it supports

CTOs, enterprise architects, open-source offices, procurement, legal, assurance, and continuity teams.

Determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

The worked scenario is an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. It is illustrative and deliberately bounded. Replace it with the organisation's actual workflow, systems, data, sources, policies, roles, infrastructure, commercial conditions and consequences while preserving the evidence discipline.

The guide's concrete output is an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan. Use the chapter explanation pages to align language, the evidence pages to create acceptance records, and any challenge pages to test whether the boundary survives missing assumptions and failure. End in a named decision, not an attractive summary.

FOUNDATIONS

Current component guides

Current guide lens; verify source and scope.

STATUS

verify per published source

Current guide lens; verify source and scope.

REPLACE

Contract + equivalence

Current guide lens; verify source and scope.

EXIT

State + proof + runbook

Current guide lens; verify source and scope.

PRIMARY RISK

A buyer can mistake a website page lists for released repositories, public source for commercial rights, inspectability for maintainability, an export file for operational portability, or a verifier for complete independent continuation.

READ THIS FIRST · DISCIPLINE

How to read claims, evidence, responsibility, and status

The website is the only factual source for this edition. The guide adds explanation, architecture and proof requirements without inventing current product, regulatory, assurance, commercial, or customer-result facts.

Six evidence layers

A published description states a current proposition. Delivered software and documentation show what is available in a selected version. Customer configuration shows how it is used. An observed run shows one result. Independent assurance has scope, method and date. A written agreement creates commercial commitments and remedies.

Responsibility stays explicit

The relevant owners include CTO/architecture, open-source programme office, engineering and build, security/supply chain, data and records, legal/procurement/licensing, operations/continuity, component maintainers, product owner and supplier. The supplier can provide product, evidence and commitments; customer owners still decide purpose, source and policy authority, professional or domain judgement, acceptable risk, infrastructure, staffing, affected-person treatment and production reliance.

Status and strong statements

Measured claims stay tied to workload, hardware and method. Research remains research. Alignment is not certification. “Open” is verified per component and licence. Pricing is re-audited at release. Roadmap and availability are not assumed. Conflicts or ambiguity become diligence items.

LAYER	EVIDENCE	CANNOT ESTABLISH ALONE
Website	website copy + captured date	Delivery, fit, contract
Delivery	Manifest + docs + artefacts	Customer result
Configuration	Data/roles/policy/ posture	Operating outcome
Run	Raw output + target + packet	General performance
Assurance	Scoped test/report + findings	Universal security/compliance
Agreement	Rights + duties + remedy	Technical behaviour without test

Keep the layers separate in every chapter and decision record.

READING RULE

Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository.

READ THIS FIRST · CANONICAL SPOTLIGHT

The foundation map and its exit value

Thirteen component guides cover current canonical foundation pages; Kera and Jacquard require separate productresearch overview status handling.

Reed, Bindery and Winnow cover parsing and collection; BitWeave contributes compact semantic retrieval; Lattice and Numerus cover deterministic rules and numerics; Ledger, Knot, Selvedge and AION cover system events, action receipts, constrained execution transcripts and proof/certificate material; FMI, HEDL and Twin cover simulation/interchange, compact encoding and event-sourced digital-twin state.

Kera has a current product description and is treated in the product and technical books as the language/compiler product, even where other website text groups it near foundations. Jacquard belongs to research status and must not be presented as a shipping foundation merely because it appears in the wider architecture story. Release, repository, package and licence status are checked per published source.

Exit value comes from the artefacts and contracts: source and build identity, schemas and formats, product state and data, source/rule/model packages, binaries and configuration, event and evidence records, public keys and trust policy, verifiers, tests, dependencies, runbooks and rights. An organisation must actually restore and use this package.

AREA	CURRENT GUIDE RECORD	VERIFY / DECIDE
Parse / collect	Reed, Bindery, Winnow	Bytes/files/web to governed inputs
Retrieve	BitWeave	Compact semantic representation and search
Rules / numbers	Lattice, Numerus	Deterministic decision and arithmetic artefacts
Events / proof	Ledger, Knot, Selvedge, AION	Different evidence propositions
Simulation / encode / twin	FMI, HEDL, Twin	Model, compact bytes, event-sourced state
Separate status	Kera product; Jacquard research	Do not collapse website categories

Current source-edition summary. Recheck publication-sensitive facts and delivered scope at the decision date.

GUIDE-SPECIFIC PROOF

Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies.

01

OPEN FOUNDATIONS, PORTABILITY, AND EXIT

Understand the decision and its language

Define the real question, current state, responsibilities, source claims, and the artefact this guide must produce.

The same evidence discipline applies to every chapter in the section.

CHAPTER 01 · EXPLAIN

Why the foundations are open

inspectability, reproducibility, portability, collaboration, and limits.

This chapter owns a specific part of the reader's decision: inspectability, reproducibility, portability, collaboration, and limits.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "Why the foundations are open", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

Inspectability

Define the exact proposition and current authoritative source.

LENS 2

Reproducibility

Name the responsible person, system, dependency, and version.

LENS 3

Portability

Specify the evidence and how an independent reviewer checks it.

LENS 4

Collaboration

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 01 · APPLY

Turn "Why the foundations are open" into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, inspectability, reproducibility, portability, collaboration, and limits., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant source and version, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for why the foundations are open; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

CHAPTER 02 · EXPLAIN

The foundation map

place AION, Bindery, BitWeave, FMI, HEDL, Knot, Lattice, Ledger, Numerus, Reed, Selvedge, Twin, Winnow, Kera, and Jacquard in the platform story as currently described.

This chapter owns a specific part of the reader's decision: place AION, Bindery, BitWeave, FMI, HEDL, Knot, Lattice, Ledger, Numerus, Reed, Selvedge, Twin, Winnow, Kera, and Jacquard in the platform story as currently described.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "The foundation map", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

Place AION

Define the exact proposition and current authoritative source.

LENS 2

Bindery

Name the responsible person, system, dependency, and version.

LENS 3

BitWeave

Specify the evidence and how an independent reviewer checks it.

LENS 4

FMI

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 02 · APPLY

Turn “The foundation map” into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, place AION, Bindery, BitWeave, FMI, HEDL, Knot, Lattice, Ledger, Numerus, Reed, Selvedge, Twin, Winnow, Kera, and Jacquard in the platform story as currently described., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant delivered mechanism, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for the foundation map; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

CHAPTER 03 · EXPLAIN

Availability is not one thing

published repository, crate/package, source on request, research, commercial product, licence rights, and release preparation.

This chapter owns a specific part of the reader's decision: published repository, crate/package, source on request, research, commercial product, licence rights, and release preparation.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "Availability is not one thing", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

Published repository

Define the exact proposition and current authoritative source.

LENS 2

Crate/package

Name the responsible person, system, dependency, and version.

LENS 3

Source on request

Specify the evidence and how an independent reviewer checks it.

LENS 4

Research

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 03 · APPLY

Turn “Availability is not one thing” into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, published repository, crate/package, source on request, research, commercial product, licence rights, and release preparation., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant customer configuration, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for availability is not one thing; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

CHAPTER 04 · EXPLAIN

Parsing and collection

Reed, Bindery, and Winnow responsibilities and boundaries.

This chapter owns a specific part of the reader's decision: reed, Bindery, and Winnow responsibilities and boundaries.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "Parsing and collection", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

Reed

Define the exact proposition and current authoritative source.

LENS 2

Bindery

Name the responsible person, system, dependency, and version.

LENS 3

And Winnow responsibilities

Specify the evidence and how an independent reviewer checks it.

LENS 4

Boundaries

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 04 · APPLY

Turn “Parsing and collection” into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, reed, Bindery, and Winnow responsibilities and boundaries., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant observed operation, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for parsing and collection; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

CHAPTER 05 · EXPLAIN

Meaning and retrieval

The semantic middle of the AI stack is supplied by a chain of providers.

This chapter owns a specific part of the reader's decision: bitWeave and the path from content to searchable state.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "Meaning and retrieval", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

BitWeave

Define the exact proposition and current authoritative source.

LENS 2

The path from content to searchable state

Name the responsible person, system, dependency, and version.

LENS 3

Scope and limits

Specify the evidence and how an independent reviewer checks it.

LENS 4

Owner and evidence

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 05 · APPLY

Turn “Meaning and retrieval” into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, bitWeave and the path from content to searchable state., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant contract and assurance, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for meaning and retrieval; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

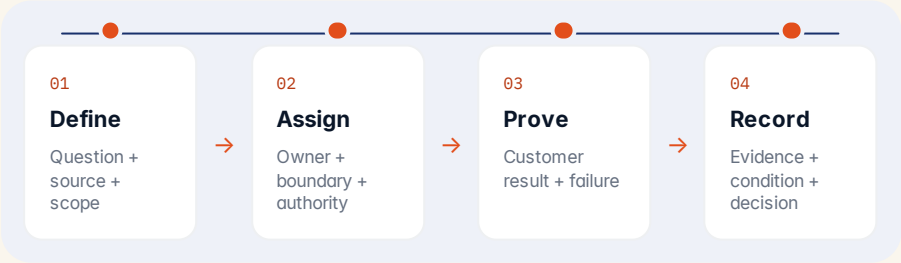
Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

02

OPEN FOUNDATIONS, PORTABILITY, AND EXIT

Define the mechanism, controls, and evidence

Turn architecture and policy language into versions, owners, boundaries, tests, records, failures, and explicit limitations.



The same evidence discipline applies to every chapter in the section.

CHAPTER 06 · EXPLAIN

Rules and numerics

Lattice and Numerus deterministic decision foundations.

This chapter owns a specific part of the reader's decision: lattice and Numerus deterministic decision foundations.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "Rules and numerics", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

Lattice

Define the exact proposition and current authoritative source.

LENS 2

Numerus deterministic decision foundations

Name the responsible person, system, dependency, and version.

LENS 3

Scope and limits

Specify the evidence and how an independent reviewer checks it.

LENS 4

Owner and evidence

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 06 · APPLY

Turn “Rules and numerics” into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, lattice and Numerus deterministic decision foundations., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant source and version, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for rules and numerics; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

CHAPTER 07 · EXPLAIN

Provenance, execution, and proof

Ledger, Knot, Selvedge, and AION; what each record proves.

This chapter owns a specific part of the reader's decision: ledger, Knot, Selvedge, and AION; what each record proves.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "Provenance, execution, and proof", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

Ledger

Define the exact proposition and current authoritative source.

LENS 2

Knot

Name the responsible person, system, dependency, and version.

LENS 3

Selvedge

Specify the evidence and how an independent reviewer checks it.

LENS 4

And AION

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 07 · APPLY

Turn "Provenance, execution, and proof" into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, ledger, Knot, Selvedge, and AION; what each record proves., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant delivered mechanism, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for provenance, execution, and proof; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

CHAPTER 08 · EXPLAIN

Simulation, encoding, and twins

FMI, HEDL, and Twin use cases and interfaces.

This chapter owns a specific part of the reader's decision: fMI, HEDL, and Twin use cases and interfaces.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "Simulation, encoding, and twins", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

FMI

Define the exact proposition and current authoritative source.

LENS 2

HEDL

Name the responsible person, system, dependency, and version.

LENS 3

And Twin use cases

Specify the evidence and how an independent reviewer checks it.

LENS 4

Interfaces

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 08 · APPLY

Turn "Simulation, encoding, and twins" into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, fMI, HEDL, and Twin use cases and interfaces., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant customer configuration, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for simulation, encoding, and twins; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

CHAPTER 09 · EXPLAIN

Language and research foundations

Kera and Jacquard status, role, and boundaries.

This chapter owns a specific part of the reader's decision: kera and Jacquard status, role, and boundaries.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "Language and research foundations", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

Kera

Define the exact proposition and current authoritative source.

LENS 2

Jacquard status

Name the responsible person, system, dependency, and version.

LENS 3

Role

Specify the evidence and how an independent reviewer checks it.

LENS 4

And boundaries

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 09 · APPLY

Turn “Language and research foundations” into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, kera and Jacquard status, role, and boundaries., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant observed operation, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for language and research foundations; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

CHAPTER 10 · EXPLAIN

How products compose the foundations

representative paths through Fabric, Loom, Nexus, Spindle, Aura, Mesh, Charter, and Core.

This chapter owns a specific part of the reader's decision: representative paths through Fabric, Loom, Nexus, Spindle, Aura, Mesh, Charter, and Core.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "How products compose the foundations", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

Representative paths through Fabric

Define the exact proposition and current authoritative source.

LENS 2

Loom

Name the responsible person, system, dependency, and version.

LENS 3

Nexus

Specify the evidence and how an independent reviewer checks it.

LENS 4

Spindle

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 10 · APPLY

Turn "How products compose the foundations" into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, representative paths through Fabric, Loom, Nexus, Spindle, Aura, Mesh, Charter, and Core., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant contract and assurance, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for how products compose the foundations; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

03

OPEN FOUNDATIONS, PORTABILITY, AND EXIT

Prove adoption, operation, continuity, and exit

Use customer evidence, challenge, recovery, diligence, commercial terms, and a signed decision before expanding reliance.

The same evidence discipline applies to every chapter in the section.

CHAPTER 11 · EXPLAIN

Reproducing a claim

clone/request source, pin version, build, run documented harness, record hardware and corpus, compare, and retain evidence.

This chapter owns a specific part of the reader's decision: clone/request source, pin version, build, run documented harness, record hardware and corpus, compare, and retain evidence.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "Reproducing a claim", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

Clone/request source

Define the exact proposition and current authoritative source.

LENS 2

Pin version

Name the responsible person, system, dependency, and version.

LENS 3

Build

Specify the evidence and how an independent reviewer checks it.

LENS 4

Run documented harness

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 11 · APPLY

Turn "Reproducing a claim" into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, clone/request source, pin version, build, run documented harness, record hardware and corpus, compare, and retain evidence., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant source and version, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for reproducing a claim; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

CHAPTER 12 · EXPLAIN

Replacing a component

contract, data/artefact format, semantic equivalence, verification, performance, and operational migration.

This chapter owns a specific part of the reader's decision: contract, data/artefact format, semantic equivalence, verification, performance, and operational migration.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "Replacing a component", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

Contract

Define the exact proposition and current authoritative source.

LENS 2

Data/artefact format

Name the responsible person, system, dependency, and version.

LENS 3

Semantic equivalence

Specify the evidence and how an independent reviewer checks it.

LENS 4

Verification

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 12 · APPLY

Turn “Replacing a component” into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, contract, data/artefact format, semantic equivalence, verification, performance, and operational migration., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant delivered mechanism, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for replacing a component; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

CHAPTER 13 · EXPLAIN

Exit package

data, source/licence material, binaries, configs, schemas, records, public keys, verifiers, build/test evidence, and runbooks.

This chapter owns a specific part of the reader's decision: data, source/licence material, binaries, configs, schemas, records, public keys, verifiers, build/test evidence, and runbooks.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "Exit package", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

Data

Define the exact proposition and current authoritative source.

LENS 2

Source/licence material

Name the responsible person, system, dependency, and version.

LENS 3

Binaries

Specify the evidence and how an independent reviewer checks it.

LENS 4

Configs

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 13 · APPLY

Turn “Exit package” into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, data, source/licence material, binaries, configs, schemas, records, public keys, verifiers, build/test evidence, and runbooks., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant customer configuration, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for exit package; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

CHAPTER 14 · EXPLAIN

Open-source diligence

licences, dependencies, maintenance, governance, vulnerability process, roadmap/status, and support.

This chapter owns a specific part of the reader's decision: licences, dependencies, maintenance, governance, vulnerability process, roadmap/status, and support.

In this guide, open foundations contribute bounded mechanisms and artefacts to commercial products. Public source, packaged release, source on request, research code, inspectability, modification rights, self-hosting, commercial licence and support are different states that must be checked per component and date. Applied to "Open-source diligence", the practical task is to name the current state, the desired state, the owner of the transition, the material input and dependency, the evidence a reviewer can inspect, and the condition that would make the claim false or the design unsuitable.

The governing boundary is this: Open code can enable inspection and reproduction while still depending on licences, packages, models, data rights, compilers, build systems, keys, operational expertise and commercial layers. Replaceability requires a stable semantic contract and migration proof, not merely another repository. Keep website publication, delivered software, customer configuration, operating result, independent assurance and written contract as separate evidence layers. A strong statement at one layer must not silently become a guarantee at another.

LENS 1

Licences

Define the exact proposition and current authoritative source.

LENS 2

Dependencies

Name the responsible person, system, dependency, and version.

LENS 3

Maintenance

Specify the evidence and how an independent reviewer checks it.

LENS 4

Governance

Record scope, failure, unresolved question, and safe next action.

OWNED ARTEFACT

This chapter contributes to an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.

CHAPTER 14 · APPLY

Turn “Open-source diligence” into decision evidence

Move from explanation to a record an accountable owner can use, challenge, update, and carry into proof, procurement, operation, or exit.

Begin with the chapter proposition, licences, dependencies, maintenance, governance, vulnerability process, roadmap/status, and support., and write it as one or more falsifiable questions. For each question, define scope and time, authoritative website source or customer source, delivered artefact, relevant observed operation, named owner, comparison or acceptance rule, raw evidence, known limitation, and decision consequence.

Use the guide-wide scenario: an architecture team pins one open foundation, reproduces its narrow claim, integrates through a declared contract, replaces it with a compatible implementation, exports product state and evidence, and restores the retained package without supplier access. Walk it end to end and ask what this chapter changes at request, source, model/tool/rule, human authority, target, evidence, deployment, operation, commercial or exit state. Do not accept a screenshot or summary when the underlying manifest, result, packet, record or contractual term can be inspected.

The proof discipline for this guide is: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies. Preserve negative and unverifiable results. A gap can be remediated, contracted, accepted by the competent owner, or block the decision; it must not disappear behind roadmap language or an average score.

QUESTION	REQUIRED RECORD	OWNER / STOP
What is claimed?	Exact wording, scope, date, source	Guide decision owner
What exists?	Delivered manifest, interface, artefact	Technical/product owner
What is configured?	Customer data, roles, rules, posture	Customer control owner
What happened?	Raw run, target state, packet, metric	Domain/operations owner
What is assured?	Test/report scope, findings, limits	Independent challenger
What is committed?	Offer/agreement, remedy, exit term	Procurement/legal owner

Evidence ladder for open-source diligence; not every question needs every row, but no row may be implied without evidence.

DECISION RECORD

Record pass, conditional pass, gap, fail, or unverifiable; link evidence, owner, remediation, date, residual risk, and effect on determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.

FIND A SUBJECT

Subject index

Page references point to the substantive explanation, worked example, control, boundary, or decision record, not merely to a passing label.

SUBJECT INDEX CONTINUED

Subject index · L–T

Terms are grouped alphabetically. Consecutive page references are collapsed into ranges.

SUBJECT INDEX CONTINUED

Subject index · T–W

Terms are grouped alphabetically. Consecutive page references are collapsed into ranges.

CLOSING DECISION RECORD

Close with one evidence-backed, reversible decision

The guide is complete when its reader can decide whether to proceed, narrow, remediate, contract a condition, hold, reject, or exit, and another person can understand why.

Assemble an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan. State the question, scope and exclusions, scenario or workflow, current baseline, canonical source snapshot, delivered manifest, customer configuration, corpus and method, results and failures, owner decisions, risks, operational and commercial conditions, evidence links, remedies, stop/exit and review triggers.

Run the final discipline: Verify current source/package/licence status for every relied-on component, build from the permitted source, run documented and customer corpus, inspect SBOM/provenance, test format and verifier, replace one boundary, export data/state/evidence, restore in clean infrastructure, and record missing rights or dependencies.

Have CTO/architecture, open-source programme office, engineering and build, security/supply chain, data and records, legal/procurement/licensing, operations/continuity, component maintainers, product owner and supplier sign only the boundaries they own. Record pass, conditional pass, gap, fail or unverifiable per proposition. Preserve dissent and raw evidence. A brochure can organise the decision; it cannot make the evidence true or take responsibility from the people who rely on it.

DECISION RECORD	open-foundations-portability-exit-decision	
QUESTION	determine what can be inspected, reproduced, pinned, self-hosted, modified, independently verified, or retained on exit.	BOUNDED
SOURCES	website copy + delivered material	CAPTURED
PROOF	corpus + failures + recovery	RUN
OWNERS	domain + technical + assurance	SIGNED
CONDITIONS	gaps + remedies + contract	DATED
DECISION	go / narrow / hold / stop / exit	RECORDED

The closing record remains useful after the presentation, release, team, supplier, or contract changes.

GUIDE OUTPUT

Produce an exit package and component ledger covering source/repository status, licence and rights, versions, build provenance, dependencies, formats, schemas, binaries, configs, data, records, keys, verifiers, tests, runbooks, support and replacement plan.