OPEN SOURCE COWBOY CONSULTING

PIONEERING OPEN SOURCE FRONTIERS

The Open Maintenance Framework (OMF): Operational Architecture for Sustaining Open Source in Web3

Authored by:
Christian Taylor
CoFounder & Chief Open Source Officer, Open Source Cowboy Consulting

Contributors:

Terence "Tex" McCutcheon
Senior Open Source Program Manager, Intersect

Sebastian Pabon
Open Source Committee Member, Intersect & CoFounder of Andamio Platform

Patrick Sheridan
Board Member, Modus Create / Tweag

Diane Mueller
Managing Director, Research and Advisory Services, Bitergia
Open Source Advisor, Hedera

Georg Link, Ph.D.
Open Source Strategist / Director of Sales, Bitergia
Co-Founder / Board member, CHAOSS Community

Christian Grobmeier
Vice President Data Privacy, Apache Software Foundation

Silona Bonewald
Founder / President, Leadingbit Solutions

Hart Montgomery
CTO, Linux Foundation Decentralized Trust

Heather Meeker
Open Source Licensing Specialist & Co-Founder of the Chinstrap Community

Andrew Aitken
Web3 & AI Open Source Strategist; Founder of the Builder Bureau

Date:
March 7, 2026

Author's Note: *This paper is meant to be read alongside the d(OSPO) paper. The two frameworks complement each other but can also be used independently. The Appendices herein simply list existing models and options in use today, not "must use" mandates for the OMF to operate.*

Executive Overview

Web3 ecosystems are uniquely positioned to address the open source sustainability problem. Yet most blockchain ecosystems replicate the same structural weakness found across the broader open source ecosystem: treasuries capable of funding infrastructure, but no operational architecture capable of sustainably deploying that capital to the maintainers who keep critical systems functioning.

Modern software is overwhelmingly dependent on open source. Research from Harvard's D³ Institute (Hoffman, Nagle & Zhou, 2024) shows open source software appears in 96% of commercial codebases, with an estimated supply-side value exceeding \$4.15 trillion. Blockchain protocols and decentralized applications operate entirely on top of this foundation. Despite this reliance, the systems responsible for sustaining maintainers remain fragmented, underfunded, and operationally fragile.

Several initiatives have begun addressing elements of this challenge. Germany's Sovereign Tech Agency has deployed more than €24.6 million to over 60 open source projects through milestone-based contracts. The Open Source Endowment is working toward a \$100 million perpetual sustainability fund. GitHub's Secure Open Source Fund and Sentry's maintainer stipend program have demonstrated measurable improvements in maintenance outcomes through structured financial support. Ethereum's Protocol Guild has shown that continuity funding can operate as a decentralized commons resistant to capture. Across the Web3 ecosystem, treasury-funded maintainer support programs are increasingly being explored.

These initiatives address funding availability. What remains largely absent is a portable operational architecture capable of translating governance mandates and funding decisions into sustained, accountable maintenance programs. The Open Maintenance Framework (OMF) is designed to fill that gap.

OMF is a portable operational architecture for sustaining decentralized open source infrastructure. It is designed to operate independently of any specific governance structure and can be executed by foundations, DAOs, elected committees, or independent operators. In ecosystems that adopt coordination bodies such as a Decentralized Open Source Program Office (dOSPO), OMF functions as the execution layer. In ecosystems without such coordination structures, OMF can be implemented directly through any accountable governance authority.

OMF is intentionally governance-neutral and blockchain-agnostic. It synthesizes operational patterns emerging from multiple independent experiments into a unified architecture that ecosystems can adopt to sustain critical open source infrastructure.

What This Paper Contributes

Portable operational architecture	A concrete, implementation-ready framework for open source sustainability in Web3 ecosystems — governance-agnostic, treasury-agnostic, and organizationally neutral.
Portfolio stewardship discipline	OMF's core innovation: a dependency-graph-driven model that allocates sustainment funding by systemic risk, not application volume. Ensures the most critical infrastructure is funded regardless of its visibility or advocacy.

Funding instrument taxonomy	Three funding instruments (Delivery, Continuity, Shared Infrastructure) × three source categories (Treasury, Endowment, Corporate Pool) — with explicit instrument-to-program mapping.
Institutional legitimacy framework	Four structural safeguards (Governance Authorization, Operator Replaceability, Maintainer Autonomy, Public Transparency) that enable OMF to function as neutral infrastructure stewardship.
Minimum viable specification	A defined minimum implementation — dependency audit, governance authority, sustainment program, reporting, portfolio review — that answers the question: what actually qualifies as OMF?

Who This Document Is For

This governance framework document is written for Web3 ecosystem designers, DAO governance participants, blockchain foundation operators, and open source community leaders responsible for the long-term sustainability of decentralized infrastructure. It is grounded in comparative analysis of real-world initiatives and is designed to be adopted, not merely read.

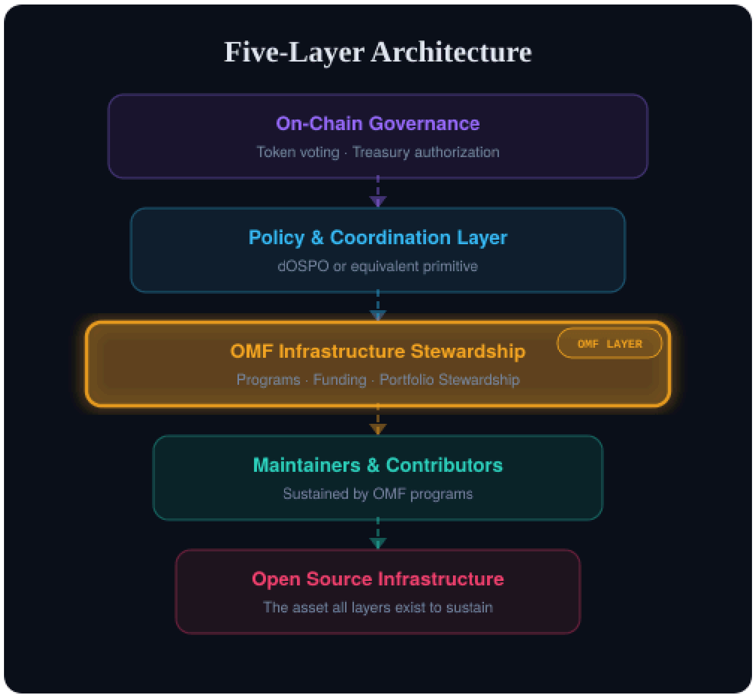
FRAMEWORK SUMMARY

Framework Summary

The Open Maintenance Framework (OMF) and the [Decentralized Open Source Program Office (dOSPO)]() together constitute a complete institutional model for governing and sustaining open source infrastructure in decentralized ecosystems. This page provides a 30-second orientation to the system.

Governance Primitive **dOSPO**	Defines how ecosystems coordinate and govern open source infrastructure — policy domains, governance bodies, execution coordination structure, and the operational layer mandate.
Operational Framework **OMF**	Defines the programmatic mechanisms through which coordination bodies execute ecosystem stewardship — maintainer sustainment, contributor pipelines, funding instruments, and infrastructure lifecycle support.
Program Portfolio	Maintainer Retainers · Code Bounties · Contributor Pathways · Operational Support · Incubation · Resilience Programs
Funding Instruments	Delivery-Oriented · Continuity-Oriented · Shared Infrastructure Support
Legitimacy Safeguards	Governance Authorization · Operator Replaceability · Maintainer Autonomy · Public Transparency

The Architecture in One Diagram


Five-Layer Architecture
On-Chain Governance
Token voting · Treasury authorization
Policy & Coordination Layer
dOSPO or equivalent primitive
OMF Infrastructure Stewardship
OMF LAYER
Programs · Funding · Portfolio Stewardship
Maintainers & Contributors
Sustained by OMF programs
Open Source Infrastructure
The asset all layers exist to sustain

SECTION 1

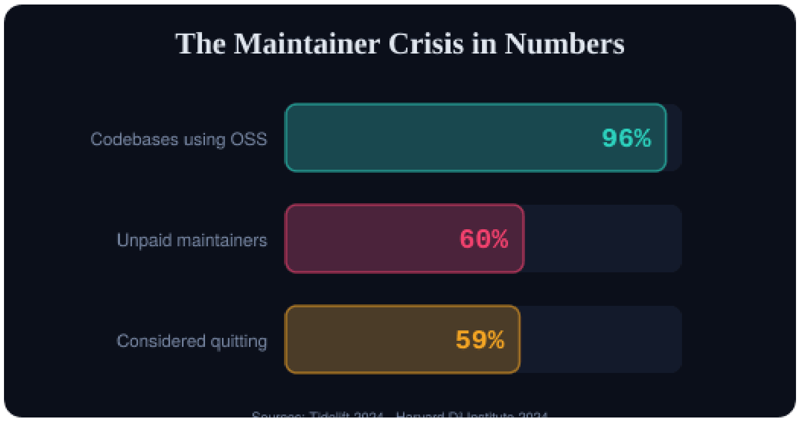
The Web3 Maintainer Crisis

Web3 ecosystems face a compounded version of the broader OSS maintainer crisis. Not only do they depend on the same foundational open source libraries as the rest of the software world — cryptographic primitives, networking stacks, data serialization formats — they also produce their own layer of critical OSS: smart contract languages, consensus implementations, wallet libraries, indexers, RPC nodes, and developer tooling. Both layers are vulnerable to the same structural failure: rapid ecosystem growth outpacing maintainer operational capacity.

RESEARCH FINDING

The technology infrastructure of the world is heavily dependent on a relatively small number of projects. A critical group of projects and their maintainers occupy an outsized role. Vulnerabilities in their projects may cause massive global disruption.

— Linux Foundation Research, Open Source Maintainers Report, 2023



The Web3-Specific Dimension

Web3 adds unique pressures on top of the general maintainer sustainability problem. Token price volatility creates feast-or-famine funding cycles. Treasury assets are denominated in native tokens, meaning that the real-dollar value of committed funding can collapse between approval and disbursement. Governance processes, while well-designed for strategic decisions, are poorly adapted to the operational cadence required for ongoing maintainer support programs.

The Documented Costs of Inaction

Maintainer Burnout	59% of maintainers have considered quitting; 60% work unpaid. The Kubernetes Ingress NGINX project announced no further security patches after March 2026 — because its maintainers burned out. (Tidelift 2024; Caceres 2025)
Security Fragility	OpenSSL operated on ~$2K/year before Heartbleed exposed global infrastructure. Undermaintained projects accumulate technical debt and unresolved vulnerabilities. (Bradbury / The Register, 2021)

Knowledge Loss	When maintainers leave, institutional knowledge of architecture and design decisions is permanently gone. No succession plan means no recovery — especially critical in Web3 where codebases are often novel and documentation sparse.
Treasury Waste	Web3 treasuries frequently fund new feature development while ignoring maintenance of what already exists. The result is compounding technical debt on infrastructure that governance participants assume is stable.

SECTION 2

The Landscape of Existing Solutions

A meaningful set of initiatives now exists addressing OSS sustainability from different angles — public sector, corporate, community-endowment, and blockchain-native. Understanding these initiatives — their models, their strengths, and their gaps — is essential context for understanding what OMF adds. The following comparative analysis draws on official sources, whitepapers, and primary documentation from each initiative.

Overview of Major Initiatives

Initiative	Funding Model	Governance	Scale	Key Risk	Operational Layer?
Sovereign Tech Agency (Germany)	Public budget; milestone contracts	GmbH; Supervisory Board; expert committees	>€24.6M to 60+ projects	Annual budget approval; procurement law	**Yes**
Open Source Endowment (USA)	Charitable endowment; ~5% spend rate	501(c)(3); board + member governance	~$750K raised; goal $100M	Long runway to scale; market risk	**Partial**
GitHub Secure OSS Fund	Corporate pool; cohort grants + training	Managed by GitHub; 14 corporate sponsors	$1.25M / 125 projects (~$10K each)	Security-only; short-duration; company priorities	**No**
Ethereum Protocol Guild	On-chain vesting; time-weighted allocation	~200 contributors; self-governed registry	Ongoing; custody-free disbursement	Ethereum-specific; eligibility consensus	**Partial**
Web3 treasury programs(Cardano, Gitcoin, etc.)	On-chain treasury; community quadratic grants	Governance-authorized committees; DAO rounds	>$60M Gitcoin; Cardano POSM Q3 2025	Token volatility; governance approval latency	**No**

Sovereign Tech Agency — The Public Model

Germany's Sovereign Tech Agency (STA) is the most mature public-sector model. Launched in 2022 by the Federal Ministry for Economic Affairs (BMWK) and administered as a GmbH under SPRIND, it has provided over €24.6 million in funding to more than 60 open source projects globally — including cURL, Drupal, and core Linux libraries — through milestone-based contracts typically ranging from €50K to €1M. (Interoperable Europe Portal, 2024)

OFFICIAL STATEMENT

The Sovereign Tech Fund is a public investment initiative launched by the German Federal Ministry... Its main goal is to strengthen digital sovereignty by supporting the maintenance of open digital base technologies — foundational open source components such as libraries, protocols, and development tools.

— Sovereign Tech Agency Official Site, 2026

Open Source Endowment — The Perpetual Fund Model

The Open Source Endowment (OSE), launched in 2026 as a US 501(c)(3), takes inspiration from university endowment models. With a target of $100M invested in a low-risk portfolio at a ~5% annual spend rate, it aims to fund critical OSS maintainers from investment returns rather than from direct donations. As of early 2026, OSE has raised approximately $750K from 96 donors, including founders from GitHub and HashiCorp. (TechCrunch, 2026)

The endowment model has important advantages: it is perpetual, not dependent on annual grant cycles, and structurally insulated from individual donor withdrawal. Its limitation is the long runway to meaningful scale — reaching $100M in assets will take years of sustained fundraising and investment returns.

GitHub Secure OSS Fund — The Corporate Cohort Model

GitHub's Secure Open Source Fund, launched in November 2024 with $1.25M from GitHub and 14 corporate partners, provides $10K grants to 125 projects that complete a 3-week security training cohort. It combines financial support with education, mentorship, and tooling. (GitHub Blog, 2024)

This model is explicitly security-focused and time-bounded. It is not a general maintenance program — it is a structured intervention to improve security hygiene across a cohort of widely-used projects. Its corporate backing and structured curriculum represent real innovation in the delivery of OSS support, but its narrow scope and one-time nature limit its impact on the broader sustainability challenge.

Ethereum Protocol Guild — The Decentralized Commons Model

Ethereum's Protocol Guild provides one of the most structurally novel models in the landscape. It maintains an on-chain registry of approximately 200 core Ethereum protocol contributors, with funding vested over four years and allocated based on time-weighted contributions. Critically, donations from ecosystem projects flow directly through the vesting contract — the Guild takes no custody of funds. There is no institutional intermediary in the chain from donor to contributor. (Protocol Guild, 2023–2025)

The Protocol Guild demonstrates that continuity funding can be structured as a decentralized, transparent commons — resistant to capture without requiring a foundation or governance body to intermediate. This design pattern is directly relevant to Web3 OMF implementations: the on-chain registry, time-weighted allocation, and custody-free disbursement are all architectural primitives OMF can adapt.

Web3 Treasury Experiments — Emerging Blockchain-Native Models

Several Web3 ecosystems have begun experimenting with treasury-funded maintainer support programs. Gitcoin Grants has deployed over $60M through community-driven quadratic matching since 2019, demonstrating that on-chain capital can reach OSS projects at scale. (Gitcoin, 2022) Cardano's treasury-funded maintainer support program, approved through governance in Q3 2025, establishes ongoing financial support for infrastructure contributors financed through on-chain treasury withdrawal actions. (Cardano POSM, 2026) These experiments validate that blockchain treasury mechanisms can fund maintainer sustainability — and surface the governance and operational design challenges any similar program must address.

SECTION 3

What the Landscape Is Missing

Taken together, the initiatives surveyed in Section 2 represent significant progress. Public investment (STA), perpetual endowment (OSE), corporate cohort programs (GitHub), decentralized commons (Protocol Guild), and blockchain treasury experiments represent real and meaningful advances. But each addresses a specific slice of the problem — and none provides a comprehensive, portable operational architecture that any ecosystem can adopt.

The Gaps in the Current Landscape

No portable framework	Every initiative is institution-specific. STA is German government. OSE is a Delaware nonprofit. Protocol Guild is Ethereum-specific. Web3 treasury programs are chain-specific. No initiative provides a portable operational architecture that any ecosystem can adopt without rebuilding from scratch.
Funding ≠ Operations	Every initiative addresses funding mechanisms. None provides a comprehensive operational architecture — the programs, processes, and accountability structures required to translate funding into sustained maintainer outcomes. Funding without operational structure produces sporadic results.
Governance gaps	On-chain governance is designed for strategic decisions, not operational coordination. Blockchain treasury experiments surface this directly — governance bodies can authorize payments but cannot manage the operational cadence of ongoing maintainer support. An operational layer that sits between governance and maintainers is required in every ecosystem.
Short-termism	Cohort programs are time-bounded. Grant rounds are episodic. Even STA operates on annual parliamentary cycles. Ongoing maintainer responsibilities require ongoing, predictable support — not interventions. Only the endowment model is structurally permanent, and it remains far from scale.
Missing contributor pipeline	None of the surveyed initiatives includes a structured contributor pathway program. CNCF's mentorship programs, which produced 25 new maintainers since 2020, demonstrate this is tractable — but it requires explicit operational design that no funding initiative alone provides.

The common failure across existing approaches is structural: funding mechanisms exist, but operational architectures do not. Ecosystems have built mechanisms to authorize spending, but not institutions capable of coordinating sustained infrastructure stewardship. The Open Maintenance Framework addresses this missing layer — defining the operational architecture through which governance decisions and capital become structured programs that sustain open source infrastructure over time.

The Synthesis OMF Provides

The landscape reveals a clear pattern: multiple ecosystems and institutions have independently converged on similar operational insights — maintainer retainers, contributor pipelines, shared tooling support, evidence-based renewal. STA's milestone contracts, the Protocol Guild's vesting registry, Cardano's treasury programs, CNCF's mentorship flywheel, and Sentry's maintainer fund all reflect variations of the same underlying operational logic.

OMF synthesizes these convergent insights into a single portable architecture. It is not derived from any single initiative — it is the generalization of patterns that multiple independent experiments have validated. Any Web3 ecosystem, regardless of chain, treasury mechanism, or governance structure, can implement OMF without dependency on or license from any specific prior program.

SECTION 4

Introducing the Open Maintenance Framework

The Open Maintenance Framework (OMF) is a structured operational architecture designed to support and sustain open source maintainers within decentralized and multi-stakeholder ecosystems. It functions as an implementation architecture — connecting governance decisions and treasury capital to the practical, ongoing work of infrastructure maintenance.

OMF is designed to stand alone. It can be executed by foundations, DAOs, elected committees, independent operators, or coordination bodies of any form — without dependency on any specific governance structure, chain, or treasury mechanism. Where ecosystems have adopted complementary governance models such as the Decentralized Open Source Program Office (dOSPO), OMF functions as the execution layer that coordination body runs. Where they have not, OMF deploys directly through whatever accountable governance structure the ecosystem already possesses.

What OMF Is — and Is Not

OMF IS	A portable, standalone operational architecture. A portfolio of modular maintainer programs. A governance-aligned accountability structure. A framework any ecosystem can adopt without precondition.
OMF IS NOT	A governance system or DAO. A foundation or legal entity. A funding source. A replacement for on-chain governance. A derivative of any specific ecosystem's program. Dependent on dOSPO or any other framework.

OMF operationalizes the insight that the sustainability crisis in open source is not primarily a funding problem — it is an operational layer problem. Funding exists, or can be created through treasury mechanisms. Governance exists. What is missing is the structured operational architecture that translates resources and priorities into sustained, effective infrastructure stewardship.

This diagnosis is consistent with how practitioners frame the challenge. As analyst Cisco Caceres concludes: "Money doesn't solve it. What burned-out maintainers need is organizational support — review help, release management, community moderation. Recognition as real work." (Caceres, 2025) OMF provides the architecture to deliver exactly this kind of structured operational support, at ecosystem scale, through governance-accountable programs.

CONVERGENT VALIDATION

Multiple independent experiments — Protocol Guild's vesting registry, Sentry's $500K maintainer fund, CNCF's mentorship flywheel (25 maintainers since 2020), and several Web3 treasury programs — have converged on the same operational insight: organized, program-based approaches to maintainer support produce measurable outcomes. OMF synthesizes these convergent patterns into a single portable architecture.

— Protocol Guild 2023–2025; Sentry 2023; CNCF 2025; Tidelift 2024

What Makes OMF Portable

Portability is OMF's central claim and it requires precise definition. OMF is portable across five dimensions:

Governance-agnostic	OMF works with DAOs, foundations, hybrid governance systems, elected committees, or multi-sig operators. It does not require a specific governance model — only that a governance body with authority to authorize programs and funding exists.
Treasury-agnostic	Compatible with on-chain token treasuries, traditional endowments, corporate pool funding, or any combination of the three source categories defined in Section 12. OMF defines funding instruments; it does not specify funding sources.
Organizationally neutral	Can be operated by any accountable organization — a foundation subsidiary (STA pattern), an independent nonprofit (OSE pattern), an embedded committee, or a multi-sig operator. The operator is replaceable; the framework is not.
Modular by program	Each OMF program — Retainers, Bounties, Pathways, Operational Support, Incubation, Resilience — can be implemented independently. Ecosystems begin with one or two programs and expand. No program is a prerequisite for another.
No institutional dependency	OMF does not require any specific ecosystem entity to exist. It does not require dOSPO. It does not require a foundation. It requires only: governance authority, a funded program budget, transparent reporting, and a portfolio review cadence.

Minimum Viable OMF

For portability to be meaningful, the minimum viable implementation must be precisely defined. An ecosystem has implemented OMF if it has all five of the following:

1. Dependency audit capability	The ecosystem has conducted a systematic dependency audit — identifying its critical OSS dependencies, scoring centrality, and publishing results. This is the foundational input to all portfolio decisions.
2. Program governance authority	A governance body has formally authorized at least one OMF program — with a defined budget envelope, selection criteria, and renewal cadence. Programs operate within this authorization; they do not self-authorize.
3. At least one sustainment program	At least one active program is sustaining infrastructure maintainers: a Retainer program, a Contributor Pathway cohort, or equivalent. The ecosystem is actively supporting people, not just studying the problem.
4. Transparent reporting mechanism	Program activities, funding allocations, and basic outcomes are published — on-chain, on a public forum, or through verifiable off-chain records — at minimum on a quarterly basis.
5. Portfolio review cadence	The funded portfolio is reviewed against the dependency map at least annually. Coverage gaps are identified. Funding decisions are updated in response to the actual dependency risk profile, not just legacy commitments.

Portability as Core Design Goal

The operational challenges facing maintainers are broadly similar whether the ecosystem is Cardano, Polkadot, Ethereum, Filecoin, or any other network. A portable framework allows ecosystems to benefit from shared design knowledge — and from the operational improvements discovered by early adopters — without requiring each ecosystem to rebuild the architecture from scratch. This is why portability is not merely a design preference; it is the primary mechanism through which OMF delivers value at ecosystem scale.

SECTION 5

Architectural Principles

OMF is grounded in six architectural principles. These function as design constraints — defining what the framework must do, what it must avoid, and how it must navigate competing pressures. Every program structure and operational mechanism within OMF should be evaluated against these principles. They are informed directly by the lessons of the initiatives surveyed in Section 2.

Governance Alignment	All OMF programs operate within the governance systems of the ecosystems they serve. For Web3 ecosystems, this means programs operate within parameters authorized by on-chain governance. Operational coordinators execute within authorized boundaries and remain accountable through reporting. Where a dOSPO or equivalent coordination body exists, OMF programs operate as its execution layer — governed by, not independent of, the ecosystem's governance framework.
Operator Neutrality	Operational coordinators do not control the infrastructure projects they support. Maintainers retain full decision-making authority over their projects. OMF provides resources and coordination, not direction. This directly guards against the "vendor capture" risks that arise when corporate sponsors (like GitHub's fund) have interests that may not align with project needs.
Replaceability	Operational operators remain replaceable through governance. No single operator, organization, or individual becomes an irreplaceable dependency. This structural safeguard addresses a key governance risk: the STA model partially avoids this through public accountability, but corporate and foundation models are more vulnerable. Replaceability must be designed in, not assumed.
Maintainer-First Design	All programs are designed with the needs of maintainers as the primary consideration. Support must reduce operational burden, not add to it. This principle is echoed by CHAOSS Project guides, which explicitly recommend moving non-code tasks off maintainers. (Foster, 2024) The GitHub cohort model demonstrates both the potential and the risk: security training adds value but also adds time demands.
Transparency	Funding allocations, program activities, and outcomes remain publicly visible to the ecosystem. Transparency is the primary accountability mechanism in decentralized governance — and is required by OpenSSF for any OSS coordination body. For Web3 ecosystems, on-chain treasury disbursements already provide a baseline of financial transparency; OMF extends this to operational outcomes.
Modular Implementation	Ecosystems can adopt individual OMF programs incrementally. The STA demonstrates the value of starting modestly and scaling: from €3.5M in 2022 to €17M planned for 2025. Modular adoption reduces implementation risk and allows programs to be validated before full deployment — critical in governance environments where failed programs create political backlash.

Learning from the Landscape

Each principle reflects a specific lesson from the comparative landscape. Governance alignment is validated by every ecosystem program that has succeeded — and violated by every one that drifted from its mandate. Operator neutrality is the risk that GitHub's corporate model must manage. Replaceability is the structural gap in OSE's early-stage single-founder governance. Maintainer-first design is where quadratic grant rounds struggle; optimizing for community voting, but also for high-level strategic decision-making by institutions such as founding entities, at the risk of undermining the legitimacy of community voting, rather than actual maintainer needs. The next section refers to the subject. Transparency is what all the surveyed initiatives share, to their credit. Modular adoption is the STA's key operational insight: start modestly, validate, then scale.

SECTION 6

Institutional Legitimacy

Operational frameworks responsible for stewarding open source infrastructure must maintain legitimacy with the ecosystems they serve. Without legitimacy, even well-designed operational programs risk being perceived as centralized control points or political instruments rather than neutral infrastructure support systems.

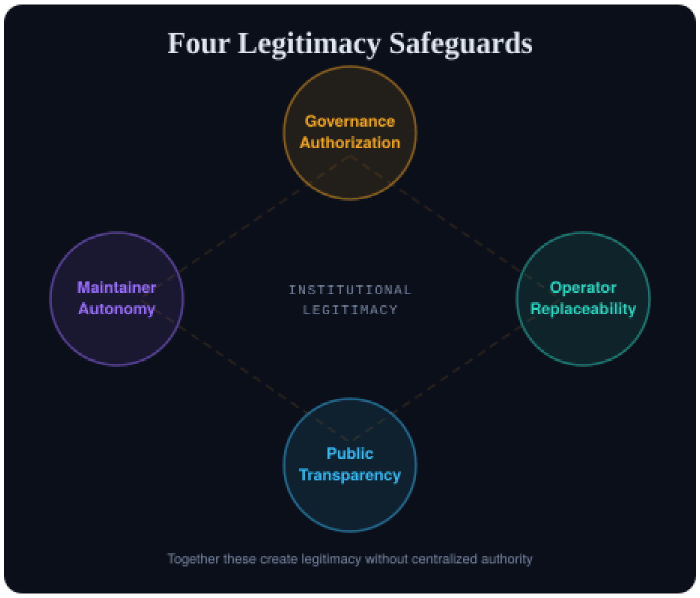
Historically, successful open source institutions have achieved legitimacy through structural safeguards rather than through reputation alone. The Apache Software Foundation relies on meritocratic project governance. The Linux Foundation emphasizes vendor neutrality across competing corporate stakeholders. The Cloud Native Computing Foundation separates technical governance from operational program management to prevent institutional capture. Ethereum's Protocol Guild achieves legitimacy through on-chain transparency and custody-free disbursement — no intermediary holds funds.

Web3 ecosystems introduce additional legitimacy challenges. Governance authority is distributed among token holders, developers, operators, and community participants. Operational programs must therefore demonstrate neutrality and accountability across multiple stakeholder groups simultaneously — including those with competing economic interests in specific infrastructure outcomes.

LEGITIMACY PRECEDENT

Major open source institutions achieve legitimacy not through reputation but through structural design: vendor neutrality, meritocratic governance, transparent operations, and separation of funding from technical authority. OMF applies these structural principles in a Web3-native context.

— Apache Software Foundation; Linux Foundation; CNCF; Protocol Guild



The Four OMF Legitimacy Safeguards

Governance Authorization	All OMF programs operate within parameters authorized by ecosystem governance. Operational coordinators execute programs but do not independently determine ecosystem priorities or funding allocations. Governance bodies retain authority to modify, suspend, or replace programs if they cease to serve ecosystem needs. This is the foundational safeguard — without it, operational programs become autonomous institutions accountable to no one.
Operator Replaceability	Operational coordinators are intentionally replaceable. No single operator, organization, or individual should become an irreplaceable dependency for the ecosystem's infrastructure stewardship mechanisms. Replaceability ensures that operational frameworks remain accountable to the ecosystem rather than the reverse. An operator that cannot be replaced has captured its program.
Maintainer Autonomy	Maintainers supported through OMF programs retain full control over their projects. Financial or operational support does not grant program operators authority over project roadmaps, technical direction, or governance structures. This separation preserves the independence that has historically defined successful open source communities — and prevents the dynamic where funding creates a reporting relationship where none should exist.
Public Transparency	All program activities, funding allocations, and ecosystem health metrics remain publicly visible. Transparency enables decentralized oversight from the broader community and reduces the risk of program capture by any single stakeholder group. For Web3 implementations, on-chain treasury disbursements provide a baseline of structural transparency that OMF extends to operational outcomes.

Together these safeguards create institutional legitimacy without requiring centralized authority. The framework remains accountable to governance, maintainers retain independence, operators remain replaceable, and the ecosystem retains visibility into outcomes. This legitimacy model allows OMF to function as neutral infrastructure stewardship rather than institutional control.

The stewardship framing is deliberate. Operations implies administration. Stewardship implies governance responsibility — the sustained, accountable management of assets held in trust for the ecosystem. OMF operates within the infrastructure stewardship layer: not as an administrator of programs, but as a steward of the infrastructure commons the ecosystem depends on.

SECTION 7

Ecosystem Architecture

Open ecosystems — whether traditional OSS communities or Web3 protocols — share a common multi-layer structure. Understanding this structure is essential to understanding where OMF fits and why its position matters.

INFRASTRUCTURE STEWARDSHIP DEFINED

Infrastructure stewardship refers to the organized coordination of maintenance, sustainment funding, and contributor pipelines for shared open source infrastructure. It is distinct from technical governance (which sets standards and direction) and from on-chain governance (which authorizes spending). Stewardship is the operational layer that translates governance decisions into sustained infrastructure outcomes.

— OMF Framework Definition

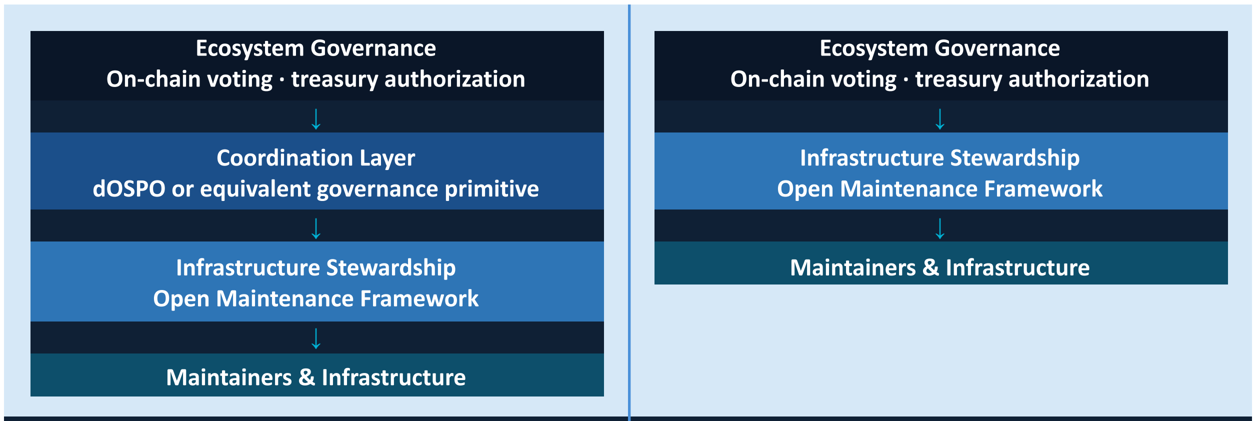
The Five Layers

On-Chain Governance	Defines strategic priorities, authorizes treasury spending, and establishes binding decisions. In Web3: DAO votes, on-chain proposals, token holder governance. Strong at strategic authorization; weak at operational coordination.
Policy & Coordination Layer	Non-binding coordination, community standards, technical committees. In Web3: foundation boards, elected governance bodies, and — in ecosystems that adopt it — the dOSPO coordination layer. Bridges on-chain decisions and operational execution.
Infrastructure Stewardship Layer	Translates governance decisions and funding into organized, accountable programs. THIS IS WHERE OMF OPERATES — and where the landscape gap is most acute. The stewardship layer does not govern; it executes governance intent through structured programs.
Maintainer Layer	Individuals and small teams maintaining specific infrastructure projects. Execute the work that keeps infrastructure functional. The ultimate beneficiaries of OMF programs.
Infrastructure Layer	The actual software, protocols, libraries, and tools the ecosystem depends on. The asset that all layers above exist to sustain.

OMF Governance Integration Models

Because OMF is governance-agnostic, it integrates into existing ecosystem architectures in two primary patterns. Both are valid. The choice depends on whether the ecosystem has adopted a formal coordination layer between governance and operations.

Model A — With Coordination Layer	Model B — Direct Integration


Ecosystem Governance
On-chain voting · treasury authorization
Coordination Layer
dOSPO or equivalent governance primitive
Infrastructure Stewardship
Open Maintenance Framework
Maintainers & Infrastructure
Ecosystem Governance
On-chain voting · treasury authorization
Infrastructure Stewardship
Open Maintenance Framework
Maintainers & Infrastructure

Ecosystems with a formal dOSPO or equivalent coordination layer. OMF executes as the operational function of that coordination body.

Ecosystems without a formal coordination layer. OMF deploys directly, operated by a foundation committee, multisig, or independent operator accountable to governance.

Both models are valid OMF implementations. The coordination layer in Model A provides additional governance infrastructure — it is not a prerequisite for OMF. The minimum requirement in both models is identical: a governance body with authority to authorize programs, a funded budget, transparent reporting, and a portfolio review cadence.

> *OMF is not a middleman — it is a structural interface. It translates governance intent into operational action, and translates operational outcomes into data that governance can use to make better resource allocation decisions. It performs this function with or without a formal coordination layer sitting above it.*

SECTION 8

Program Architecture

These programs represent the operational portfolio that the ecosystem's coordination operator executes. Governance authorizes their scope and funding; OMF defines their design and execution. Each addresses a specific dimension of infrastructure sustainability and is validated by real-world precedents from the comparative landscape. Programs can be implemented independently or in combination, allowing ecosystems to begin with one or two and expand as capacity develops.

Maintainer Retainer	Ongoing financial support tied to maintenance responsibilities rather than specific deliverables. Multiple ecosystems have independently arrived at this mechanism: Sentry's fund, Protocol Guild's vesting model, and various Web3 treasury programs all implement variants. Tidelift (2024) finds paid maintainers produce significantly more maintenance improvements than unpaid ones.
Code Bounties	Funded bounties for specific features or bug fixes. Used across the Gitcoin ecosystem, GitHub's security cohort, and various Web3 treasury grant programs. Complements retainers by incentivizing targeted development work without requiring ongoing commitment.
Contributor Pathways	Structured mentorship and onboarding programs. Directly modeled on CNCF's LFX Mentorship flywheel: 25 mentees became project maintainers since 2020. (CNCF, 2025) The most evidence-backed mechanism for growing the future maintainer pipeline.
Operational Support	Administrative and coordination services: security audit facilitation, legal review, documentation support, governance assistance. Reduces non-code overhead for maintainers without requiring them to source these services individually.
Incubation Program	Structured support for early-stage critical projects. Helps emerging infrastructure reach maturity without burning out founding maintainers before the project attracts co-maintainers. STA's milestone contracts for emerging projects provide a strong operational model.
Resilience Programs	Succession planning, dependency audits, knowledge documentation. Specifically addresses the bus-factor failures that have taken down critical infrastructure — including the Kubernetes Ingress NGINX shutdown. (Caceres, 2025)

The Full System

While each program can be implemented independently, they are most effective as a coherent system. Contributor pathways grow the future maintainer pool that sustains the retainer program. Operational support reduces maintenance burden, making retainer support go further. Resilience programs preserve the institutional knowledge that retainer programs help accumulate. This portfolio is precisely what the infrastructure stewardship layer executes — the programs governance authorizes, OMF defines.

SECTION 9

Maintainer Sustainment Mechanisms

Maintainer sustainment is OMF's core program. The design of sustainment mechanisms must balance meaningful support with preserving maintainer independence. Tidelift's 2024 survey confirms the mechanism works: paid maintainers make significantly more code and maintenance improvements than unpaid ones. Sentry's 2023 fund distributed $500K to over 500 maintainers, observing that even modest stipends "bring great motivation." (Whitacre / Sentry, 2023)

Why Maintenance Requires Different Mechanisms Than Development

Development work is project-bounded: clear beginning, defined deliverables, defined end point. Maintenance work is ongoing and indefinite: it continues as long as the project is in use. This distinction has profound implications for support design. Treasury grants and bounties — the dominant Web3 support mechanisms — are optimized for development work. They create episodic support for ongoing responsibilities.

This mismatch is precisely why several ecosystems have independently invented the maintainer retainer — a support structure tied to ongoing availability rather than deliverable completion. It is also why the Sovereign Tech Agency uses milestone-based contracts rather than one-time grants — creating predictable, ongoing funding tied to continued maintenance work. The convergence of these independent designs on the same pattern is strong evidence that the pattern is correct.

Sustainment Mechanism Options

Maintainer Retainers	Regular support tied to ongoing maintenance responsibilities. Funded through treasury withdrawals (Cardano model) or operational program budgets. Does not create employment; preserves maintainer independence while providing baseline stability.
Infrastructure Credits	Direct provision of computing, storage, or tooling resources. Reduces operational overhead without creating financial dependency relationships. GitHub's fund partially implements this through tooling access for cohort participants.
Operational Support Services	Access to shared administrative resources: security audit facilitation, legal review, compliance documentation. Reduces non-code burden without requiring each maintainer to individually source these services. Modeled on the structured support components of GitHub's Secure Fund training cohort.
Coordination Support	Assistance with community management and communication work. The Linux Foundation blog identifies community management as a specific area where dedicated support frees technical maintainers. Reduces social overhead that contributes heavily to burnout.

Sustainment mechanisms must reduce the operational burden on maintainers, not add to it. A sustainment program that requires more administrative work than the support it provides is *counterproductive. This is the most common failure mode in well-intentioned OSS support programs.*

SECTION 10

Contributor Pathways

Healthy Web3 ecosystems require a continuous pipeline of contributors growing into maintainers. Without structured pathways, maintainer programs eventually exhaust their pool — and no amount of retainer funding prevents an ecosystem from collapsing if no new maintainers are being cultivated.

The Evidence Base

CNCF's mentorship flywheel provides the strongest available evidence for structured contributor pathways. Through LFX Mentorship, Google Summer of Code, and Outreachy integration, CNCF has documented that 25 mentees became project maintainers since 2020, with many participants making multiple contributions and advancing to reviewer and approver roles. (CNCF, 2025) This is not theoretical — it is documented operational outcome data.

> **KEY OUTCOME DATA**
>
> *CNCF mentorship flywheel: 25 mentees became maintainers since 2020. Programs include LFX Mentorship, Google Summer of Code, and Outreachy. Participants continue contributing after completion and progress to reviewer and approver roles across cloud-native infrastructure.*
>
> — CNCF Blog, The Mentorship Flywheel, 2025

The CHAOSS Project's contributor sustainability metrics reinforce this: "contributor sustainability has a large impact on overall project sustainability," and most projects struggle to find enough people to maintain them long term — particularly projects with a single maintainer. (Foster, 2024)

Pathway Design for Web3

Web3-specific contributor pathways face unique design challenges. The technical barrier to entry is higher than in most OSS domains. Cryptographic correctness requirements are unforgiving. Developer tooling ecosystems are less mature. And the economic incentives of Web3 attract contributors motivated by token appreciation rather than long-term infrastructure stewardship — a fundamentally different contributor profile from what sustainable infrastructure maintenance requires.

Effective Web3 contributor pathways must therefore include not only technical onboarding but also community integration and values alignment — helping contributors understand the long-term stewardship orientation required for infrastructure maintenance versus the sprint-oriented orientation common in protocol development.

Web3 Contributor Pathway Precedents

Several Web3 ecosystems have developed native contributor pathway programs that OMF can learn from and build upon. These are not exhaustive, but they represent the most structured approaches currently operating:

Ethereum Protocol Fellowship	A structured program onboarding developers into Ethereum core protocol development — the closest Web3 equivalent to CNCF's LFX Mentorship. Participants receive direct mentorship from core contributors and work on real protocol improvements. This is the model for OMF Contributor Pathway design at the protocol layer.

Polkadot Fellowship	A merit-based technical collective managing the Polkadot/Kusama protocol — with explicit tiered membership (rank 0–9) and clear progression criteria. Demonstrates that formalized contributor advancement structures are viable in decentralized ecosystems. OMF's tiered contribution model is directly informed by this approach.
Gitcoin Grants / Quadratic Funding	Provides financial support to OSS contributors through community-validated quadratic funding rounds. While not a structured mentorship program, it demonstrates that community-driven contributor recognition mechanisms can operate at ecosystem scale — a useful complement to OMF's program-based pathways.
CNCF LFX Mentorship (adapted)	25 mentees became maintainers since 2020. While not Web3-native, its flywheel model — structured pairing, stipend support, defined contribution milestones — is directly applicable to any Web3 ecosystem. OMF treats this as the reference implementation for contributor pathway design.

- Define tiered contribution levels with explicit criteria for progression
- Create structured first-contribution experiences giving new contributors early wins
- Pair new contributors with experienced maintainers for mentorship
- Document institutional knowledge to reduce expertise barriers
- Recognize contributor milestones to reinforce long-term engagement

SECTION 11

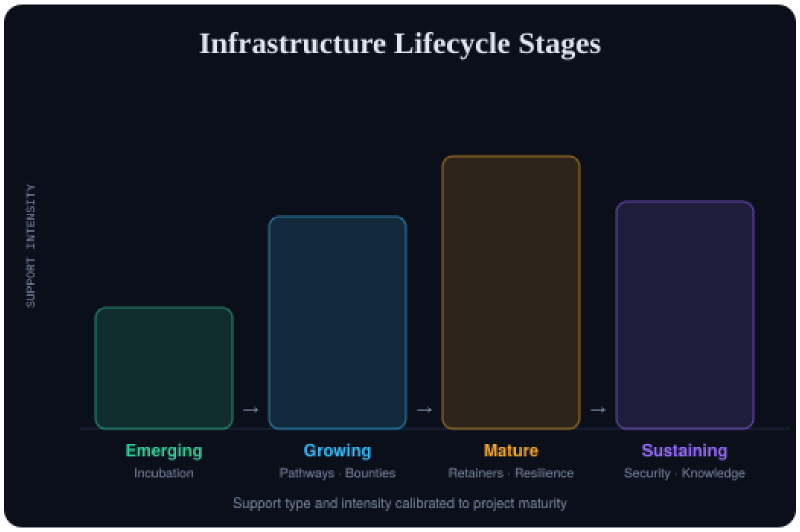
Infrastructure Lifecycle Support

Open source infrastructure projects evolve through recognizable lifecycle stages, and the support needs of a project differ substantially depending on its stage. OMF's lifecycle support program allocates resources based on actual stage-specific needs rather than applying uniform support across all projects.

Some Examples include:
1) CNCF: Project Lifecycle and Process | CNCF Contributors
2) Intersect's MBO Incubation Cycle: Project Incubation Lifecycle Framework | Intersect - Open Source Committee | Intersect Committees
3) Historical Hyperledger (now LFDT) Process: Project Lifecycle | Hyperledger TOC

Lifecycle Stages

Emerging	Early-stage projects with unproven adoption. Primary needs: technical mentorship, community building, basic infrastructure support. Primary risk: founder burnout before the project reaches sufficient adoption to attract co-maintainers. Incubation program applies here.
Growing	Projects experiencing rapid adoption increase. Primary needs: scalable maintainer support, contributor onboarding, coordination capacity. Primary risk: the gap between adoption speed and operational capacity — the most common trigger for maintainer burnout in successful Web3 projects.
Mature	Widely adopted, stable projects with established communities. Primary needs: ongoing maintenance support, security responsiveness, succession planning. Primary risk: maintainer complacency about succession until a crisis forces the issue.
Sustaining	Long-term critical infrastructure with slow active development. Primary needs: security-focused maintenance, dependency management, knowledge documentation. Primary risk: gradual neglect as projects become "too boring" to attract new contributors despite being mission-critical.



Stage-Appropriate Allocation

The STA's approach offers a useful model here: its milestone-based contracts are designed to match the specific operational needs of each funded project rather than applying a one-size-fits-all grant. OMF formalizes this into a lifecycle assessment process — periodically reviewing the stage of each supported project and calibrating the type and level of support accordingly.

For Web3 ecosystems, lifecycle-aware support is particularly important because the pace of ecosystem adoption can be extremely rapid. A library that is an "emerging" project in one governance cycle can be a "growing" dependency for thousands of applications in the next. OMF programs must be designed to respond to this pace.

SECTION 12

Funding Architecture for Web3

Funding is where Web3 ecosystems most clearly expose the mismatch between governance ideals and operational reality. Treasuries are often substantial, yet critical infrastructure remains under-maintained. OMF reframes funding as a governance function rather than a series of discretionary payouts — structuring it through defined instruments, explicit principles, and transparent cost accounting.

This section defines three OMF funding instruments drawn from decentralized open source practice, four governing principles that apply across all instruments, and a cost-of-coordination framework that makes operational overhead explicit and accountable. Together these form the OMF funding architecture — a model any Web3 ecosystem can implement regardless of its specific treasury mechanism or governance structure.

Why Existing Approaches Fail

Before defining what OMF proposes, it is worth naming precisely why the approaches most Web3 ecosystems currently rely on are structurally insufficient for maintainer sustainability.

Grants	Reward proposal articulation over stewardship. End abruptly when deliverables are met, regardless of ongoing dependency. A project that is critical infrastructure the day after grant completion receives no support to remain so.
Bounties	Incentivize discrete tasks but fragment ownership. Do not fund availability, incident response, or the ongoing operational responsibility that distinguishes a maintainer from a contributor. A bounty hunter is not a steward.
Donations	Fluctuate with community sentiment, privileging popular projects over critical ones. The most important infrastructure is often the least visible — making donation-based funding structurally biased against the projects that need it most.
Ad hoc committee coordination	Distributes coordination cost invisibly across committees, ad hoc coordinators, and duplicated tooling — creating the illusion of zero overhead while accumulating hidden inefficiencies that surface only during crises.

Stakeholder research across Web3 ecosystems has documented this mismatch directly: being overly rigorous about quantifiable metrics needs to match the maturity of those receiving funds. Separate framework specifications have proposed tiered accountability structures with differentiated verification requirements at micro-grant, growth, and infrastructure levels — rather than applying uniform rigor regardless of context. (Consensus Consulting, M1 Stakeholder Interviews and M2 Framework Specifications)

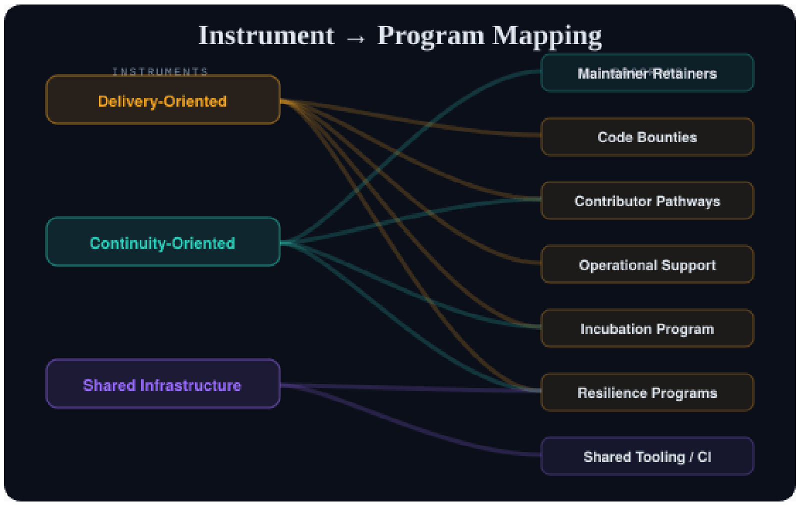
The Four OMF Funding Principles

OMF structures all funding decisions according to four principles. These apply regardless of which funding instrument is used or which funding category provides the capital. They are design constraints — not aspirations.

Lifecycle Alignment	Funding instruments must match project maturity and dependency risk. A library in early development requires different support than a decade-old critical dependency. Applying the same instrument to both wastes resources on one and under-serves the other.
Time-Bounded Support	All funding has explicit duration. Continuity is earned through renewal, not assumed by default. This prevents the accumulation of legacy commitments that crowd out emerging critical projects, and creates structured accountability checkpoints.
Evidence-Based Renewal	Renewal decisions reference outcomes, not advocacy. Projects seeking continued funding must demonstrate stewardship outcomes — security posture, contributor health, dependency reliability — not simply make the case that they deserve support. This discipline is what separates a program from a patronage system.
Portfolio Discipline	Funding decisions consider ecosystem-wide dependency coverage and opportunity cost. No individual project decision is made in isolation — every commitment is evaluated against the full portfolio of infrastructure the ecosystem depends on and the total funding available to sustain it.

The Three OMF Funding Instruments

OMF coordinates three classes of funding instruments. Each is defined precisely — with its scope, appropriate lifecycle stage, and verification approach — to prevent the instrument-misuse patterns that cause existing approaches to fail.



Instrument 1: Delivery-Oriented Funding

DEFINITION

Delivery-oriented funding supports scoped, verifiable work: features, documentation, testing improvements, security remediations. Funding is tied to explicit deliverables with defined completion criteria. Most effective in early and growth stages where discrete improvements are the primary need.

— OMF Instrument Definition · informed by comparative landscape analysis

Delivery-oriented funding is the most familiar instrument and the easiest to verify — which is also why it is overused. Its strength is accountability: a deliverable is either shipped or it is not. Its weakness is that it cannot fund availability. A maintainer who responds to a security incident at 2am on a Saturday is not delivering a discrete feature — they are fulfilling an operational responsibility that no bounty or grant can be structured to compensate.

OMF uses delivery-oriented funding for: new contributor onboarding improvements, documentation that reduces maintenance burden, testing infrastructure that prevents regressions, and time-bounded security remediation sprints. It is explicitly not the instrument for ongoing maintenance responsibility. GitHub's Secure Open Source Fund ($10K per project against defined security milestones) is a well-designed implementation of delivery-oriented funding at the security layer. (GitHub Blog, 2024)

Instrument 2: Continuity-Oriented Funding

DEFINITION

Continuity-oriented funding supports long-lived, high-dependency infrastructure through maintainer retainers or service commitments. It compensates availability and operational responsibility — without conferring ownership — and is most appropriate for projects that have become critical ecosystem dependencies.

— OMF Instrument Definition · Protocol Guild model; Tidelift 2024

Continuity-oriented funding is the instrument most Web3 ecosystems currently lack and most urgently need. It is structurally different from all grant-based approaches: rather than paying for work done, it pays for the capacity to do work — the ongoing availability that makes a maintainer a steward rather than a contractor.

Early treasury-funded maintainer continuity programs have demonstrated that governance-authorized retainers can provide stable, long-term support for critical infrastructure contributors. Tidelift's 2024 research validates the mechanism — paid maintainers produce measurably more maintenance improvements than unpaid counterparts, and 60% of OSS maintainers currently work entirely unpaid.

Protocol Guild — The Continuity Funding Precedent

Ethereum's Protocol Guild provides the most compelling existing precedent for continuity-oriented funding as a commons. It maintains an on-chain registry of approximately 200 core Ethereum protocol contributors, with funding vested over four years and allocated based on time-weighted contributions. Donations from ecosystem projects flow directly through the vesting contract — the Guild takes no custody of funds. (Protocol Guild, 2023–2025)

WHY PROTOCOL GUILD MATTERS FOR OMF

The Protocol Guild demonstrates that continuity funding can be structured as a decentralized, transparent commons — resistant to capture, without requiring an institutional intermediary. Most Web3 ecosystems currently lack an equivalent mechanism for funding critical infrastructure maintainers. OMF provides the architecture to build one.

— Protocol Guild · Ethereum Protocol Guild Documentation, 2023–2025

The Protocol Guild model has direct implications for OMF design. Its on-chain registry, time-weighted allocation, and vesting-based disbursement are all patterns OMF can adapt for ecosystem-specific implementations. The absence of an institutional intermediary in the custody chain is a critical design feature — it reduces the capture risk that plagues most corporate and foundation funding models.

Instrument 3: Shared Infrastructure Support

DEFINITION

Shared infrastructure support funds tooling consumed across multiple projects as a public good: CI systems, SDKs, security tooling, dependency management infrastructure. These are projects that every other project depends on — but whose beneficiaries are diffuse, making them systematically underfunded by per-project grant mechanisms.

— OMF Instrument Definition · informed by STA and OSE models; Consensus Consulting

Shared infrastructure is the most systematically underfunded category in Web3 ecosystems. Per-project grant programs fund the projects they can identify. Shared tooling — the CI runner that every project uses, the SDK that every dApp depends on, the security scanner that runs across the entire ecosystem — has no clear grant applicant because its beneficiaries are everyone and its owner is effectively no one.

OMF addresses this through portfolio-level funding decisions: the shared infrastructure instrument is explicitly portfolio-scoped rather than project-scoped. The ecosystem governance body authorizes funding for tooling that serves the portfolio, not for a specific project's roadmap. Germany's STA demonstrates this logic: its funding targets "open digital base technologies — foundational open source components such as libraries, protocols, and development tools" — infrastructure whose value is multiplied precisely because it is shared. (Interoperable Europe Portal, 2024)

Instrument-to-Program Mapping

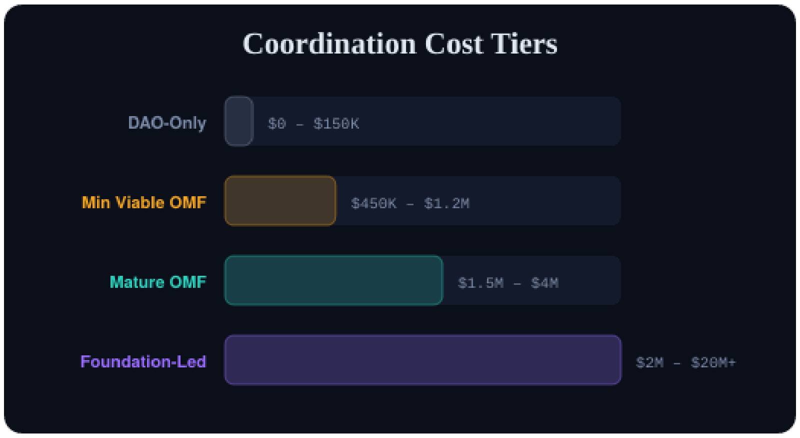
The three instruments map to OMF's program architecture with deliberate specificity. Matching instrument to program is as important as selecting the right funding source — mismatches are the primary cause of well-funded programs that still fail to sustain maintainers.

OMF Program	Delivery-Oriented	Continuity-Oriented	Shared Infra
Maintainer Retainer	—	✓ **Primary**	—
Code Bounties	✓ **Primary**	—	—
Contributor Pathways	✓ **Primary**	✓ Supplement	—
Incubation Program	✓ **Primary**	✓ Transition	—
Resilience Programs	✓ Audits	✓ Succession	✓ Tooling
Shared Tooling / CI	—	—	✓ **Primary**

Cost of Coordination

Coordination is not free — but failing to acknowledge its cost leads to hidden inefficiencies that are worse than the overhead itself. OMF makes coordination cost explicit, benchmarked, and accountable. The common objection that a structured operational framework adds overhead misses the point: much of this cost already exists, distributed invisibly across committees, ad hoc coordinators, duplicated tooling, and crisis response. OMF consolidates that distributed cost and makes it visible, optimizable, and governed.

DAO-only coordination(minimal operators)	~$0 – $150K annually. Zero institutional overhead. High execution risk: coordination depends entirely on volunteer availability and informal accountability. Suitable only for very early-stage ecosystems with minimal infrastructure dependency.
Foundation-led stewardship	$2M – $20M+ annually depending on scope. Reflects full institutional overhead: legal standing, executive leadership, compliance, infrastructure hosting. Appropriate for ecosystems with mature legal requirements and substantial operational complexity.
Minimum viable OMF(program lead, PM, part-time security, analytics)	~$450K – $1.2M annually. Provides structured program management, accountability infrastructure, and basic security coordination without full institutional overhead. The entry point for most Web3 ecosystems moving beyond ad hoc coordination.
Mature OMF(multiple programs, dedicated security, technical PM)	$1.5M – $4M annually. Supports a full portfolio of active programs across all three funding instruments, with dedicated security coordination and technical program management. Appropriate for ecosystems with significant infrastructure dependency and treasury capacity.



Funding Source Categories

OMF instruments can be funded from multiple source categories. The three categories below each have distinct structural properties that affect which instruments they can most reliably support — and which risks they introduce. The most resilient OMF implementations draw from all three.

Treasury & Protocol-Native	Continuity retainers, Code Bounties
Endowment & Perpetual Fund	Long-term Retainers, Contributor Pathways

Corporate & Philanthropic Pool	Delivery programs, Incubation, Security cohorts

Governance Authority Over All Funding Is Non-Negotiable

Regardless of instrument or source category, governance retains authority over all OMF funding allocation decisions. OMF programs execute within governance-authorized envelopes — they do not set their own budgets. Treasury withdrawals require on-chain approval. Endowment disbursements require board authorization aligned with ecosystem governance. Corporate pool allocations require transparent, publicly documented criteria that governance has reviewed and approved.

Operational programs that control their own funding become structurally independent of governance. This independence may look like efficiency in the short term but becomes a source of misalignment and accountability risk over time. Multi-source diversification strengthens sustainability; governance authority over all categories ensures the funding serves the ecosystem rather than the programs.

SECTION 13

Operational Governance

Operational governance refers to the mechanisms through which OMF programs remain accountable to the ecosystems they serve. It is distinct from on-chain governance: where on-chain governance makes strategic decisions, operational governance ensures programs execute in alignment with those decisions and remain transparent about outcomes.

Mechanisms

- Regular public reporting on program activities, resource allocation, and outcomes — published on-chain or via verifiable off-chain records
- Periodic program reviews with governance body involvement, including explicit sunset criteria for programs that are not delivering outcomes
- Clear escalation paths for governance intervention when programs drift from authorized parameters
- Public documentation of sustainment criteria and selection processes to prevent program capture
- Auditable financial records for all disbursements — essential in Web3 contexts where treasury spending is already publicly verifiable

Lessons from the Comparative Landscape

The STA's governance model — a Supervisory Board providing strategic oversight with an expert team handling day-to-day operations — is a strong model for the embedded implementation of OMF. It separates governance accountability from operational execution while maintaining clear lines of authority.

Some blockchain ecosystems have implemented committee-based operational governance structures that execute within parameters authorized by on-chain governance — handling operational detail off-chain without requiring full community votes on every program decision. This separation of governance authority from operational execution is a design pattern OMF explicitly adopts: governance authorizes the envelope; operators execute within it.

Corporate-governed funds, by contrast, are governed entirely internally, with sponsors having implicit influence over program direction. This is appropriate for a corporate initiative but would be problematic for an ecosystem-wide operational framework — which is precisely why OMF's governance alignment principle requires accountability to ecosystem governance, not operational sponsors.

GOVERNANCE PRINCIPLE

Like roads and bridges, open source software makes up critical infrastructure that many depend on. Insufficient investment and maintenance lead risk to accumulate over time. Long-term, consistent support for OSS — as for roads and bridges — is generally preferable to waiting for catastrophic failure.

— OpenSSF, Why Open Source is Infrastructure, 2023

SECTION 14

Portfolio Stewardship

Funding an individual project is an allocation decision. Funding an ecosystem's infrastructure is a portfolio discipline. OMF introduces portfolio-level thinking as a core operational function — the systematic management of funded programs as an interconnected portfolio of ecosystem dependencies, not a collection of independent grants.

This distinction is foundational. Grant programs fund the applications they receive. Portfolio stewardship actively maps ecosystem dependencies, identifies coverage gaps, and allocates funding toward the infrastructure that poses the highest systemic risk if it fails — whether or not that infrastructure has an articulate advocate writing proposals.

PORTFOLIO RISK EVIDENCE

Harvard's D³ Institute (2024) found that 96% of commercial codebases depend on OSS with fewer than 10 contributors. The risk is not distributed evenly — it is concentrated in a small number of high-centrality dependencies. Portfolio stewardship discipline targets exactly this concentration.

— Hoffman, Nagle & Zhou, Harvard D³ Institute, 2024

Portfolio Stewardship Functions

Dependency mapping	Systematic identification of all infrastructure the ecosystem depends on — direct dependencies, transitive dependencies, and foundational libraries. Reveals the actual dependency graph rather than relying on projects that self-identify as critical.
Centrality scoring	Assessment of how many ecosystem components depend on each infrastructure piece. High-centrality dependencies represent concentrated systemic risk — a single failure point for many dependent projects. These receive prioritized sustainment consideration.
Coverage gap analysis	Comparison of the dependency map against current OMF program coverage. Identifies critical infrastructure with no program support — the highest-risk gap in the ecosystem's sustainability posture.
Portfolio rebalancing	Periodic review of the funded portfolio against updated dependency maps. As the ecosystem evolves, new dependencies emerge and old ones deprecate. Portfolio discipline means the funding portfolio evolves with the dependency reality, not with historical momentum.
Sunset discipline	Explicit criteria for program termination or renewal. Time-bounded commitments prevent legacy programs from crowding out emerging critical infrastructure. Every renewal is an active portfolio decision, not a default continuation.
Opportunity cost accounting	Every funding commitment displaces alternatives. Portfolio stewardship makes opportunity cost explicit — governance can see not just what is funded, but

	what is not, and why. This prevents the invisible accumulation of technical sustainability debt.

The Dependency Centrality Metric

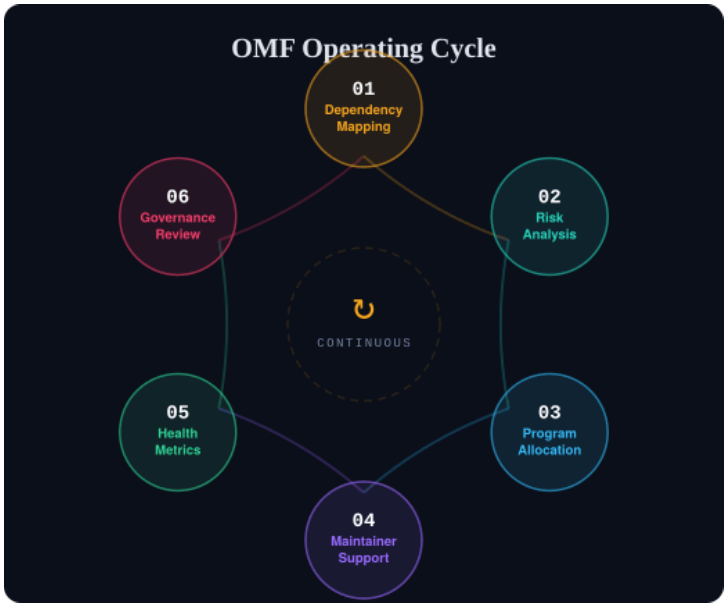
Dependency centrality, the count of ecosystem projects that depend on a given library or component, is the primary risk metric in OMF portfolio stewardship. High-centrality, low-sustainability infrastructure is the highest-priority sustainment target: maximum impact if it fails, minimum visibility in standard grant programs.

This metric is used in Linux Foundation research, CHAOSS dependency risk analysis, and OpenSSF's criticality scoring work. Accurate centrality scoring depends on complete, machine-readable dependency data, which is why OMF treats Software Bills of Materials (SBOMs) in SPDX or CycloneDX format as the essential data substrate for portfolio stewardship. SBOMs make dependency graphs auditable, repeatable, and portable across projects; without them, centrality scoring is only as reliable as the manual inventory it draws from. (Appendix A provides the full dependency audit methodology, including SBOM generation guidance and tooling for Web3 technology stacks.) OMF adopts centrality as a portfolio prioritization input — not as the sole criterion, but as the primary risk lens for coverage gap analysis. A project with 1,000 dependents and one maintainer who reports burnout is a portfolio emergency. A project with 10 dependents and three healthy maintainers is not.

Portfolio stewardship is what separates OMF from all prior funding initiatives. STA funds what its expert committee recommends. GitHub funds a security cohort. Protocol Guild funds Ethereum protocol contributors. OMF funds what the ecosystem's dependency graph says is at risk — systematically, transparently, and with governance authority over every decision.

The OMF Operating Cycle

Portfolio stewardship is not a one-time exercise — it is a recurring governance cycle. The OMF operating cycle runs continuously, with each phase feeding into the next. The cycle is the mechanism that keeps the funding portfolio aligned with actual ecosystem dependency risk over time.



The cycle has no natural termination point — it runs as long as the ecosystem depends on open source infrastructure. Each governance review produces an updated portfolio allocation that feeds directly into the next dependency mapping cycle, closing the loop between evidence and action.

SECTION 15

Implementation Models

OMF can be implemented through three structural models depending on ecosystem characteristics, governance maturity, and existing organizational infrastructure. All three are represented in the comparative landscape — meaning each model has real-world validation, not just theoretical justification.

Model 1: Embedded (Foundation-Integrated Pattern)

In the embedded model, OMF operations are carried out by staff or working groups directly integrated into the ecosystem's governance structure. This mirrors the STA pattern — the Agency is a SPRIND subsidiary, not an independent organization — and the approach used by several Web3 ecosystems that have embedded operational committees within broader governance frameworks.

Advantages: Direct governance integration; lower coordination overhead; aligned incentives. Risks: Difficulty maintaining operational independence from governance; potential for programs to become politically captured by governance priorities. Mitigation: Explicit separation of operational and governance roles within the committee structure.

Model 2: Independent Operator (OSE / Sentry Pattern)

In the independent operator model, OMF operations are carried out by an organization structurally separate from the ecosystem's governance body — authorized to execute programs within defined parameters. The Open Source Endowment operates this way: it is legally independent of any specific ecosystem, receiving funding from diverse donors and allocating it according to its own criteria. Sentry's fund similarly operates as an internal program independent of any blockchain governance.

Advantages: Clear separation of operational and governance functions; replaceable if quality declines. Risks: Coordination overhead; potential for misalignment over time. The replaceability principle is essential in this model — it is the primary governance safeguard.

Model 3: Hybrid (CNCF / Foundation Pattern)

The hybrid model combines embedded and independent elements. CNCF demonstrates this effectively: technical projects are governed by their own communities, the CNCF TAC provides oversight, and operational programs (like LFX Mentorship) are run by dedicated staff with their own operational autonomy. Some OMF programs (like operational support services) may work best as embedded functions while others (like an endowment-style fund) may require independent legal structure.

Most mature Web3 ecosystems will find that a hybrid implementation makes best use of existing organizational infrastructure — embedding OMF coordination within an existing foundation or governance body while spinning off specific programs (particularly endowment-style funding vehicles) as independent entities.

Suggested Adoption Path

Regardless of implementation model, ecosystems benefit from a phased adoption path. Attempting to deploy the full OMF program portfolio simultaneously overloads governance capacity and creates multiple simultaneous failure modes. The STA's own trajectory — €3.5M in 2022, scaling to €17M by 2025 — demonstrates the value of starting modestly and expanding as operational confidence grows.

Phase 1 — Pilot	Phase 2 — Expansion	Phase 3 — Full Portfolio
Maintainer Retainers (2–5 projects) Contributor Pathways (1 cohort) Dependency mapping (ecosystem audit)	Resilience Programs (succession planning) Code Bounties (targeted sprints) Infrastructure lifecycle support	Shared infrastructure funding| Ecosystem health monitoring Full portfolio management

First 12 Months: A Concrete Starting Point

For a mid-sized Web3 ecosystem with a $5–20M treasury, the following first-year roadmap provides a turn-key starting point. It is designed to be governable, affordable, and demonstrably valuable within 12 months — the minimum viable OMF implementation.

Month	Action	Cost (est.)	Success Indicator
1–2	Dependency audit — map all critical OSS dependencies, score centrality, identify coverage gaps	~$15–30K (contractor)	Dependency map published; top-20 risk projects identified
2–3	Governance proposal — submit OMF pilot budget request (Maintainer Retainers + 1 Contributor Pathway cohort)	~$5–10K (facilitation)	Governance approval; budget authorized; operator appointed
3–6	Launch Retainer program — 3–5 projects, 6-month initial commitments; launch first contributor cohort	~$200–400K/yr (retainers + pathway)	Retainers active; cohort enrolled; first progress reports
6–9	Mid-point review — publish outcome data; governance review of retainer performance	~$10K (reporting)	Outcome report published on-chain or verifiable off-chain
9–12	Renewal cycle — evidence-based renewal decisions; add Resilience programs if capacity allows	~$50–100K (resilience)	Renewal decisions documented; succession plans for top-5 projects

Total first-year cost estimate for minimum viable OMF: $280–570K depending on retainer scope. Well within a $5M treasury's 5–10% operational budget. The governance proposal template, program criteria templates, and dependency audit methodology are available at opensourcecowboy.org.

SECTION 16

Measuring Ecosystem Health

Measurement is essential to adaptive management. OMF programs should be evaluated against key indicators of ecosystem health — tracked over time, published transparently, and used to inform governance decisions. The CHAOSS Project provides the most comprehensive available framework for OSS health metrics and informs this section directly. The Linux Foundation also offers LFX to its hosted projects to help keep track of general health kpis and metrics.

Key Health Indicators

Maintainer Sustainability	Maintainer attrition rate; time-in-role; self-reported burnout; compensation adequacy. CHAOSS and Tidelift surveys both measure these dimensions. Declining sustainability metrics are leading indicators of project failure — available months or years before a public incident.
Infrastructure Stability	Security vulnerability response times; release frequency; open issue backlog age; dependency compatibility. The GitHub Secure Fund's security milestone structure provides a useful operational model for security-specific metrics.
Contributor Participation	New contributor volume; contributor-to-maintainer conversion rate; time-to-first-contribution; organization diversity. CNCF's mentorship outcomes (25 new maintainers) demonstrate that these metrics reflect real pipeline health, not just activity.
Dependency Resilience	Bus factor per project; number of projects with documented succession plans; contributor concentration by organization. Directly measures the single-point-of-failure risks that have caused recent high-profile infrastructure failures.
Program Effectiveness	Stipend-to-activity correlation; pathway completion rates; bounty completion rates; treasury deployment efficiency. Essential for governance accountability — connecting funding to outcomes.
Dependency Centrality	Measures how many ecosystem projects depend on a given library or infrastructure component. High-centrality dependencies represent systemic risk — a single failure point for many dependent projects. This metric directly informs portfolio prioritization decisions: high-centrality, low-sustainability infrastructure is the highest-priority sustainment target. Used in Linux Foundation research, CHAOSS framework, and dependency risk analysis at scale.

Web3-Specific Measurement Considerations

Web3 ecosystems have unique measurement advantages and challenges. On-chain activity provides objective, verifiable data on protocol usage and treasury disbursements. But off-chain maintainer health — burnout, succession planning, institutional knowledge retention — requires qualitative measurement approaches that on-chain data cannot capture.

OMF recommends combining on-chain metrics (treasury disbursements, program utilization) with periodic off-chain surveys aligned with CHAOSS methodology, published as public ecosystem health reports. The Atlantic Council's analogy is instructive: just as bridges have public inspection reports, critical OSS infrastructure should have documented, regular health assessments — not just issue trackers.

SECTION 17

Risks, Precedents & The Road Ahead

Risk Landscape

Centralization	OMF programs becoming de facto control points. Mitigated structurally by replaceability principle and governance oversight. The STA model — public accountability through parliamentary oversight — is an instructive precedent. Any OMF implementation must preserve explicit governance authority to replace operational operators.
Program Capture	Programs captured by specific maintainers, funders, or governance participants. Mitigated by transparency requirements and publicly documented allocation criteria. The replaceability principle is the primary structural safeguard — any operator that cannot be replaced has effectively captured its program.
Token Volatility	For Web3 implementations, treasury funding in native tokens creates volatility risk. Mitigated by stable-asset conversion mechanisms, multi-source funding architecture across all three funding categories, and operating reserves. This is an active design challenge for any blockchain treasury program.
Governance Fatigue	On-chain governance participants may resist ongoing program oversight requirements. Mitigated by off-chain committee structure that handles operational detail without requiring full community votes on every decision — precisely the separation that layered governance architecture provides.
Scope Creep	Operational programs expanding beyond their authorized parameters. Mitigated by explicit sunset criteria and periodic program reviews. OMF's modular design allows scope to be explicitly authorized and periodically reconfirmed — each program renewal is an accountability checkpoint.

The Road Ahead

The comparative landscape reveals that the field is moving rapidly. Germany has committed to scaling STA funding to €17M annually. The Open Source Endowment is building toward $100M. GitHub has engaged 14 corporate partners in its security cohort. Protocol Guild continues to grow its contributor registry. Multiple Web3 ecosystems are experimenting with treasury-funded maintainer programs. The momentum is real — what is missing is the cross-ecosystem coordination and shared operational architecture that OMF provides.

Open source infrastructure does not sustain itself.

It survives because institutions emerge to support it.

Web3 ecosystems now possess capabilities previous generations of open source did not: programmable governance, transparent treasuries, and global contributor coordination. The challenge is no longer funding. It is institutional architecture.

The Open Maintenance Framework provides that architecture.

About Open Source Cowboy

The Open Maintenance Framework was developed by Open Source Cowboy (opensourcecowboy.org) as part of ongoing work on decentralized open source governance and infrastructure sustainability. OMF is designed to be adopted, adapted, and improved through use. We invite governance participants, foundation operators, and maintainers across Web3 to engage with the framework and contribute to its evolution.

Evidence Summary

Claim	Primary Sources	Confidence
OSS in 96% of commercial codebases	*Hoffman, Nagle & Zhou (Harvard D³, 2024)*	**High**
60% unpaid maintainers; 59% considered quitting	*Tidelift State of Maintainer Report (2024)*	**High**
STA: >€24.6M to 60+ projects	*Interoperable Europe Portal (2024); Sovereign Tech Agency (2026)*	**High**
OSE: ~$750K raised; $100M goal	*TechCrunch / Bort (2026); OSE Official Site (2026)*	**High**
GitHub Fund: $1.25M, 125 projects, $10K each	*GitHub Blog (2024); OSI Blog (2024)*	**High**
Protocol Guild: ~200 contributors, 4yr vesting, no custodian	*Protocol Guild documentation (2023–2025)*	**High**
Web3 treasury programs: Gitcoin >$60M; Cardano POSM approved	*Gitcoin Blog (2022); Cardano POSM News (2026)*	**High**
CNCF: 25 mentees became maintainers since 2020	*CNCF Mentorship Flywheel Blog (2025)*	**High**
Paid maintainers produce more improvements	*Tidelift (2024)*	**High**
dOSPO + OMF as complementary governance/operational stack	*Synthesized from comparative analysis; Open Source Cowboy (2025)*	**Medium**
OMF as portable framework across ecosystems	*Synthesized from comparative analysis; no single source*	**Medium**

Bibliography

Bort, J. (2026). A VC and some big-name programmers are trying to solve open source's funding problem, permanently. TechCrunch. https://techcrunch.com/2026/02/26/

Bradbury, D. (2021). Untangling open source's sustainability problem. The Register. https://www.theregister.com/2021/05/10/untangling_open_sources_sustainability_problem/

Caceres, C. (2025). Open source maintainer burnout: Critical infrastructure is dying. RoamingPigs. https://roamingpigs.com/field-manual/open-source-maintainer-burnout/

Cardano POSM (2026). The paid open source model. https://www.intersectmbo.org/news/the-paid-open-source-model

CNCF (2025). The mentorship flywheel: How CNCF is growing the next generation of cloud native leaders. https://www.cncf.io/blog/2025/12/18/the-mentorship-flywheel

Consensus Consulting (2025). M1 Stakeholder Interviews and M2 Framework Specifications. https://hackmd.io/@ConsensusConsulting

Foster, D. (2024). From data to action: Using metrics to improve open source communities. CHAOSS / OpenSource.net. https://opensource.net/chaoss_guides/

GitHub Blog (2024). Announcing GitHub Secure Open Source Fund. https://github.blog/news-insights/company-news/announcing-github-secure-open-source-fund/

Hoffman, M., Nagle, F., & Zhou, Y. (2024). The value of open source software. Harvard D³ Institute. https://d3.harvard.edu/revealing-value-the-economic-power-of-open-source-software/

Interoperable Europe Portal (2024). Funding open source: case study on the Sovereign Tech Fund. https://interoperable-europe.ec.europa.eu/collection/open-source-observatory-osor/document/funding-open-source-case-study-sovereign-tech-fund

Jahnsson, J. (2022). Gitcoin Grants to advance DEI in Web3. Gitcoin Blog. https://www.gitcoin.co/blog/gitcoin-grants-to-advance-dei-in-web3

Linux Foundation Research (2023). Open source maintainers report. https://project.linuxfoundation.org/hubfs/LF%20Research/Open%20Source%20Maintainers%202023%20-%20Report.pdf

Open Source Endowment (2026). World's First Endowment Fund for OSS. https://endowment.dev/

OpenSSF (2023). Why open source is infrastructure, and why it matters. https://openssf.org/blog/2023/03/08/why-open-source-is-infrastructure-and-why-it-matters/

OSI / Vidal, N. (2024). Improving open source security with the GitHub Secure Open Source Fund. https://opensource.org/blog/improving-open-source-security-with-the-new-github-secure-open-source-fund

Protocol Guild (2023–2025). Protocol Guild documentation and vesting contract. https://protocol-guild.readthedocs.io/

Sovereign Tech Agency (2026). Home / About. https://www.sovereign.tech/

Tidelift (2024). State of the open source maintainer report. https://assets-eu-01.kc-usercontent.com/.../2024-tidelift-state-of-the-open-source-maintainer-report-.pdf

Whitacre, C. (2023). We just gave $500,000 to open source maintainers. Sentry Blog. https://blog.sentry.io/we-just-gave-500-000-dollars-to-open-source-maintainers/

Appendix A: Dependency Audit Methodology

Supports: Section 14 (Portfolio Stewardship) and Section 16 (Measuring Ecosystem Health)

Portfolio stewardship lives or dies on the quality of the dependency map. This appendix provides the concrete methodology, tooling, and walkthrough that ecosystems need to execute the dependency audit described in the main framework. It is grounded in CHAOSS community standards and adapted for Web3-specific technology stacks.

CHAOSS Dependency Metrics Standard

The CHAOSS (Community Health Analytics for Open Source Software) project provides the most comprehensive available framework for dependency health measurement. OMF adopts three CHAOSS metrics as the foundation for dependency audit and ongoing monitoring:

Libyears	Measures the cumulative age gap between a project's dependencies and their current stable releases. Supports analysis of both direct and transitive dependencies. CHAOSS research (Cox et al. 2015) demonstrates that projects using outdated dependencies are four times more likely to have security issues, making Libyears a leading risk indicator for portfolio stewardship decisions.
Upstream Code Dependencies	Identifies all projects involved in building a given project, including organizational context, vulnerability exposure, and licensing implications downstream. This metric maps the actual dependency graph rather than relying on projects that self-identify as critical.
OSS Project Viability	An aggregate model combining dependency freshness, contributor health, release frequency, governance clarity, and community activity into a holistic viability assessment. Provides the multi-dimensional view required for portfolio prioritization beyond single-metric scoring.

CHAOSS Tooling for Dependency Analysis

The CHAOSS community maintains two primary software platforms that operationalize these metrics:

Augur	Goes beyond counting activities to include license coverage, COCOMO-based software complexity and cost-of-replacement data by project, software dependency scanning, Libyears measurement, and time-series OpenSSF Scorecard information. Includes anomaly detection and extensible data visualization through Dash/Plotly. Directly useful for OMF centrality scoring and portfolio prioritization.
GrimoireLab 2.0	Provides historical data tracking, GDPR-compliant identity management, and a business-layer integration for commercial services. Offers new capabilities for analyzing software

	development health at the project and ecosystem level. Used by Bitergia to produce repo maturity reports for multiple ecosystems.

Web3-Specific Tooling Stack

Web3 ecosystems use diverse technology stacks that require language-specific dependency scanning. The following tooling covers the primary Web3 development ecosystems:

Rust (Solana, Polkadot, NEAR, Substrate)	cargo-audit for known vulnerability scanning; cargo-deny for license, advisory, and source policy enforcement; cargo-tree for transitive dependency mapping. Cargo's lockfile (Cargo.lock) provides deterministic dependency resolution suitable for SBOM generation.
Solidity / EVM (Ethereum, L2s, EVM-compatible chains)	Slither for static analysis including dependency detection; OpenZeppelin's dependency graph tooling; Hardhat/Foundry dependency resolution. Note: Solidity's import model differs from package managers — library dependencies often require manual enumeration supplemented by automated scanning.
Go (Cosmos SDK, Tendermint-based chains)	go mod graph for full transitive dependency mapping; govulncheck for vulnerability scanning; go mod tidy for cleanup. Go modules provide strong dependency resolution with verifiable checksums (go.sum).
Haskell (Cardano core)	cabal-plan-bounds and stack for dependency resolution; cabal freeze for deterministic builds. Haskell's package ecosystem (Hackage) is smaller but dependency chains can be deep.
TypeScript/JavaScript (dApp layers, SDKs, tooling)	npm audit / yarn audit for vulnerability scanning; npm ls for dependency tree; Dependabot or Renovate for automated updates. The JavaScript ecosystem has the highest transitive dependency density of any major language ecosystem.

Handling Non-Scannable Dependencies

Not all Web3 infrastructure dependencies appear in package manifests. Ecosystems must supplement automated scanning with a declared dependency process for:

Proprietary or closed-source indexers	Infrastructure like block explorers, indexing services, and data providers that the ecosystem depends on but cannot scan. Require self-declaration by operators or ecosystem survey.
Oracle networks and bridge infrastructure	Cross-chain dependencies that introduce systemic risk but exist outside any single ecosystem's package graph.
RPC providers and hosted infrastructure	Centralized services that large portions of the ecosystem route through. Dependency is operational rather than code-level, but the risk is equivalent.

Forked or vendored libraries	Code copied into repositories rather than imported as dependencies. Invisible to standard dependency scanners. Requires SBOM generation at build time to capture.

SBOM as the Data Format Layer

Software Bills of Materials provide the standardized data format for making dependency audit results portable and machine-readable. Two dominant standards exist:

SPDX (Linux Foundation)	ISO/IEC 5962:2021. Strong in license compliance and provenance tracking. Preferred by Microsoft and Google. Supports JSON, XML, YAML formats. Version 3.0 adds profiles for Security, Build, AI, and Dataset use cases.
CycloneDX (OWASP)	Designed for security use cases: vulnerability identification, VEX (Vulnerability Exploitability Exchange), and dependency analysis. Lightweight, JSON/XML/Protocol Buffer formats. Current version 1.7.
Package URL (purl)	Supported by both SPDX and CycloneDX. Standardizes component identification across ecosystems using a universal naming convention (e.g., pkg:cargo/serde@1.0.197). Essential for cross-ecosystem dependency matching.

Contributor Health as a Leading Dependency Risk Indicator

Dependency risk is not only a function of code age and vulnerability exposure. CHAOSS contributor sustainability metrics — contributor attrition rate, time-in-role, organizational diversity, and progression through contribution ladders — are leading indicators of dependency risk. A project whose contributor pipeline is drying up is a dependency risk even if the code looks healthy today. Ecosystems implementing structured contributor progression frameworks (such as tiered contribution ladders with explicit advancement criteria) can use progression rates and mentorship activity as early warning signals for portfolio stewardship decisions.

Audit Walkthrough: The First 60 Days

The following walkthrough describes what a dependency audit looks like in practice for a mid-sized Web3 ecosystem. This is the operational foundation for OMF's minimum viable implementation (Section 4).

Weeks 1–2: Automated scanning	Run language-specific dependency scanners across all ecosystem-critical repositories. Generate SBOMs in SPDX or CycloneDX format. Aggregate into a unified dependency graph. Identify all direct and transitive dependencies.
Weeks 3–4: Manual supplement	Conduct declared dependency survey for non-scannable infrastructure (oracles, indexers, RPC providers, vendored code). Interview maintainers of top-20 most-used repositories to identify undocumented dependencies and operational dependencies.

Weeks 5–6: Centrality scoring	Apply PageRank-style centrality scoring to the dependency graph. Weight by: number of dependent projects, depth of dependency chain, and whether the dependency is direct or transitive. Cross-reference with CHAOSS contributor health data for each dependency.
Weeks 7–8: Publication	Publish the dependency map, centrality scores, and coverage gap analysis. Identify top-20 highest-risk dependencies (high centrality, low sustainability). Present to governance for portfolio allocation decisions. Establish baseline metrics for ongoing monitoring.

References: CHAOSS Community, chaoss.community/metrics; Cox et al. 2015, Measuring Dependency Freshness in Software Systems; OpenSSF Scorecard, scorecard.dev; SPDX, spdx.dev; CycloneDX, cyclonedx.org

Appendix B: OMF Program Operations Specification

Supports: Section 8 (Program Architecture), Section 9 (Maintainer Sustainment Mechanisms), and Section 12 (Funding Architecture)

This appendix defines the operational requirements that OMF programs must satisfy. These are specifications, not prescriptions each ecosystem implements them according to its governance structure and organizational context. Implementation examples are drawn from multiple independent initiatives to illustrate the range of valid approaches.

Retainer Program Operational Requirements

Any OMF-compliant retainer program must define the following elements. The specific parameters are set by each ecosystem's governance; OMF requires that they be explicitly defined and publicly documented.

Role tiers	At minimum, two tiers distinguishing between senior project stewards with final technical authority and contributors focused on day-to-day maintenance, documentation, community operations, or ecosystem advocacy. Responsibilities, authority levels, and access permissions must be explicitly differentiated.
Evaluation period	A defined probationary period (OMF recommends 30 days) during which new retainer participants demonstrate capability, responsiveness, and community engagement before ongoing funding is confirmed. A self-assessment or structured report should be required at completion.
Quarterly review cadence	Evaluations at minimum every three months covering: contribution activity, security and stability practices, community engagement and mentorship, documentation quality, and alignment with ecosystem priorities. Consecutive failures must trigger warnings and potential funding suspension.
Annual renewal review	A comprehensive assessment of effectiveness, continued relevance, and succession planning. Results determine continued funding, modification of terms, or removal from the program. Every renewal is an active portfolio decision, not a default continuation.
Appeals mechanism	A structured process for maintainers to appeal negative evaluations within a defined window (OMF recommends 14 days). Appeals must be reviewed by the oversight body with clear outcomes: probation, reversal, or confirmation of the original decision.
Reapplication process	Maintainers removed from the program may reapply after a defined cooldown period (OMF recommends six months), demonstrating renewed contribution activity and improved engagement. Reapplication should require endorsement from current program participants or committee members.

Dual oversight structure	At minimum, technical review (ensuring alignment with infrastructure priorities) and governance/compliance review (ensuring adherence to open source best practices, licensing, and security standards) must be structurally separated. Both must agree on approvals, renewals, and terminations.

Sample Retainer Selection Rubric

The following weighted rubric is a recommended starting template. Ecosystems should calibrate weights to their specific priorities and publish the rubric before each selection cycle to prevent program capture.

Dependency centrality (Weight: 30%)	Number of ecosystem projects that depend on this infrastructure. Scored from the dependency audit (Appendix A). High-centrality projects represent systemic risk and receive prioritized consideration regardless of their visibility or advocacy.
Bus factor (Weight: 25%)	Number of active maintainers with deep knowledge of the codebase. Projects with bus factor of 1 receive maximum risk score. Scored from contributor data and maintainer interviews.
Maintainer capacity and burnout risk (Weight: 20%)	Self-reported maintainer availability, workload, and burnout indicators. Informed by CHAOSS contributor sustainability metrics and direct survey. Declining capacity is a leading indicator of project failure.
Security incident history (Weight: 15%)	Frequency, severity, and response time of past security incidents. Projects with unresolved critical vulnerabilities or slow response patterns receive elevated risk scores.
Governance priority alignment (Weight: 10%)	Degree to which the project aligns with ecosystem strategic priorities as defined by governance. This weight is intentionally lowest to prevent political capture of the selection process.

Renewal Decision Tree

OMF requires that renewal decisions follow an explicit, documented process rather than ad hoc judgment. The following decision tree operationalizes evidence-based renewal:

Automatic renewal recommendation	Maintainer meets all quarterly KPI thresholds for consecutive review periods. No governance flags. Recommendation proceeds to governance for confirmation without requiring full re-evaluation.
Triggered review	Maintainer falls below threshold on one or more non-critical metrics for a single review period. Formal review initiated with specific improvement targets and timeline. Continued funding during review period.
Escalated review	Maintainer falls below threshold on critical metrics (security response time, release cadence) OR fails non-critical metrics for

	two consecutive quarters. Governance body directly involved. Funding may be suspended pending review outcome.
Sunset recommendation	Maintainer fails critical metrics for two consecutive review periods despite remediation plan. OR project dependency centrality has declined below portfolio threshold. Program operator recommends termination to governance with documented rationale.

Bounty Program Operational Requirements

OMF-compliant bounty programs (delivery-oriented funding) must define:

Dual eligibility criteria	Separate qualification requirements for projects requesting funded work and developers fulfilling bounties. Projects must demonstrate ecosystem alignment and active usage. Developers must demonstrate relevant technical expertise and contribution history.
Approval-to-payment workflow	A structured sequence: project/feature approval by oversight body → bounty creation with defined scope and acceptance criteria → developer selection based on qualifications and bid quality → implementation → technical and functional evaluation → payment upon acceptance.
Post-delivery quality verification	A defined evaluation period (OMF recommends 30 days) after feature integration for community testing and feedback. Critical issues discovered during this period must be resolved by the developer before final acceptance and payment.
Performance benchmarks	Explicit requirements for feature completeness, security compliance (no introduced vulnerabilities), code quality (linting, maintainability review), test coverage, and documentation.
Reapplication timeline	Developers facing repeated rejections must wait a defined period (OMF recommends three months) before reapplying. Persistent inability to meet quality standards may result in exclusion from bounty participation.

Tooling Sustainability Program Requirements

Shared infrastructure tooling requires a distinct program structure because its beneficiaries are diffuse and it is systematically underfunded by per-project mechanisms.

Project intake checklist	Applicants must demonstrate readiness for sustainability funding by meeting baseline requirements covering: repository hygiene, governance documentation, delivery planning, and security practices. This checklist prevents sustainability funding from being used as general-purpose grants.
Ecosystem impact evaluation	A working group or evaluation committee assesses applications based on ecosystem-wide impact, community demand, usage metrics, adoption rates, and interoperability contributions. Selection prioritizes tools that serve the portfolio, not a single project's roadmap.

Working group structure	A dedicated working group focused on identifying, evaluating, and supporting mission-critical tooling. The working group defines key categories (developer experience, interoperability, security tooling, dependency management) and uses community feedback loops and data-driven analysis to inform funding decisions.
SBOM and dependency integration	Funded tooling projects should integrate Software Bills of Materials and dependency analysis tools (such as Bitergia or Augur) to contribute to ecosystem-wide dependency health monitoring.

Implementation Examples

The following initiatives each implement portions of the OMF program operations specification. No single implementation covers all requirements; taken together, they validate the specification's operational patterns:

STA milestone contracts	Delivery-oriented funding pattern. Milestone-based contracts with defined deliverables, matched to project-specific operational needs. Demonstrates that public-sector funding can operate with structured accountability.
Protocol Guild vesting registry	Continuity-oriented funding pattern. On-chain registry of approximately 200 contributors with time-weighted allocation and four-year vesting. No institutional intermediary in the custody chain. Demonstrates decentralized continuity funding as a commons.
Sentry maintainer fund	Retainer pattern. Distributed $500K to over 500 maintainers. Demonstrated that even modest stipends produce measurable motivation and maintenance improvements.
CNCF LFX Mentorship	Contributor pathway pattern. 25 mentees became project maintainers since 2020. Structured pairing, stipend support, and defined contribution milestones. The reference implementation for contributor pipeline design.
Cardano POSM (pilot)	Web3-native implementation of multiple OMF patterns. Maintainer Retainer program with two-tier roles, 30-day evaluation, quarterly reviews, and dual-committee oversight (TSC + OSC). Code For Us bounty program with structured approval-to-payment workflow. Tooling Sustainability program with working group evaluation and $1M ADA allocation. Currently in early-stage pilot.

Note: These examples illustrate the range of valid implementation approaches. OMF does not endorse any single implementation as the standard — it defines the operational specification that any ecosystem can implement according to its governance structure.

Appendix C: Cross-Chain Interoperability and Shared Dependency Standards

Supports: Section 4 (Introducing OMF — Portability), Section 7 (Ecosystem Architecture), and Section 14 (Portfolio Stewardship)

OMF claims blockchain-agnosticism and portability. For that claim to be meaningful in a fragmented Web3 world, ecosystems running OMF must be able to share dependency health data and coordinate funding for shared infrastructure without requiring cross-chain governance. This appendix defines the standards-based approach to cross-chain interoperability.

The Shared Dependency Problem

Multiple blockchain ecosystems depend on the same foundational libraries: libp2p for networking, protobuf for serialization, cryptographic primitives (curve implementations, hash functions), and widely-used development frameworks. Each ecosystem audits and funds these dependencies independently, creating duplication of effort and persistent coverage gaps for shared infrastructure that no single ecosystem considers its responsibility.

SBOM Standards as the Interoperability Layer

Rather than proposing a new protocol or shared governance structure, OMF leverages existing Software Bill of Materials standards as the data format for cross-ecosystem dependency health sharing:

SPDX	Linux Foundation project, ISO/IEC 5962:2021. The only internationally recognized SBOM standard. Strong in license compliance, provenance tracking, and organizational attribution. Version 3.0 adds profiles for Security, Build, Usage, AI, and Dataset use cases. Preferred by enterprises and government procurement.
CycloneDX	OWASP Foundation standard focused on security use cases. Native support for VEX (Vulnerability Exploitability Exchange), hashing, and dependency trees. Lightweight, designed for CI/CD pipeline integration. Current version 1.7 adds citation provenance tracking.
Package URL (purl)	Universal component identifier supported by both SPDX and CycloneDX. Standardizes naming across ecosystems: pkg:cargo/serde@1.0.197, pkg:npm/express@4.18.2, etc. Essential for matching dependencies across ecosystems that use different package managers.

SLSA Framework for Supply Chain Integrity

Supply-chain Levels for Software Artifacts (SLSA, pronounced “salsa”) provides a security framework and checklist of standards for preventing tampering, improving integrity, and securing packages and infrastructure. OMF implementations can adopt SLSA levels for verifying the provenance of funded deliverables — ensuring that code delivered under bounty programs or milestone contracts is built from verified source with documented build processes.

Proposed OMF Dependency Health Schema

OMF proposes a lightweight data schema that any OMF implementation can publish and any other can consume. This is not a protocol — it is a data standard for making dependency audit results interoperable:

Project identifier	Package URL (purl) for code dependencies; ecosystem-defined identifier for non-code infrastructure (oracles, indexers, RPC providers).
Centrality score	Number of ecosystem projects that depend on this component, as calculated during the dependency audit (Appendix A). Published as a numeric score with methodology documentation.
Maintainer count and bus factor	Number of active maintainers with commit access; number with deep architectural knowledge. Derived from CHAOSS contributor data.
Funding status	Whether the dependency is currently funded through an OMF program, and if so, which instrument (delivery, continuity, shared infrastructure). Enables other ecosystems to see what is and is not covered.
CHAOSS health metrics	Libyears score, release frequency, median PR response time, contributor diversity index. Published from Augur or GrimoireLab data.
Last audit date	Timestamp of most recent dependency audit that included this component. Enables other ecosystems to assess data freshness.

Federated Co-Funding Model

When multiple ecosystems depend on the same library, they can co-fund retainers through coordinated but independent governance decisions. This model requires no shared governance — only data visibility:

Identification	Each ecosystem's published dependency health data reveals shared dependencies. A library like libp2p appearing in Ethereum, Filecoin, and Polkadot dependency maps is visible to all three ecosystems.
Proportional contribution	Each ecosystem contributes to the retainer proportionally to its dependency centrality on that library. An ecosystem where libp2p has centrality score 500 contributes more than one where it has centrality score 50.
Independent governance	Each ecosystem's governance body independently authorizes its contribution. No cross-chain voting or shared treasury mechanism is required. The coordination is informational, not structural.
Transparent reporting	The funded maintainer reports to all contributing ecosystems through the shared health schema. Each ecosystem can verify that its contribution is producing outcomes.

OpenSSF Criticality Scoring

OpenSSF’s criticality scoring methodology provides a shared risk assessment approach that OMF implementations can adopt for cross-ecosystem comparability. By using a common scoring methodology, ecosystems can compare their dependency risk profiles and identify shared vulnerabilities that would benefit from coordinated funding.

References: SPDX, spdx.dev; CycloneDX, cyclonedx.org; SLSA, slsa.dev; Package URL specification, github.com/package-url; OpenSSF Criticality Score, github.com/ossf/criticality_score

Appendix D: Web3 Contributor Pathway Adaptations

Supports: Section 10 (Contributor Pathways) and Section 11 (Infrastructure Lifecycle Support)

Web3 ecosystems face unique contributor pipeline challenges: higher technical barriers to entry, cryptographic correctness requirements, less mature developer tooling, and economic incentives that attract contributors motivated by token appreciation rather than long-term infrastructure stewardship. This appendix defines OMF's contributor pathway specification for Web3 and provides adaptations for these challenges.

OMF Contributor Pathway Specification for Web3

Any OMF-compliant contributor pathway must include the following elements, adapted for Web3-specific conditions:

Tiered progression	At minimum, four stages with explicit criteria for advancement: entry-level contributor, regular committer, trusted committer with review authority, and core maintainer with strategic decision-making responsibility. Each tier defines responsibilities, access levels, and quantifiable progression milestones.
Mentorship pairing	New contributors are paired with experienced maintainers for structured guidance. Mentors help select appropriate first tasks, provide technical context, and integrate newcomers into the project's culture and decision-making norms.
Maintenance-oriented early stages	Early pathway tasks are deliberately oriented toward maintenance work: bug triage, documentation improvement, test coverage, dependency updates. This serves dual purposes — it delivers immediate value to the project and naturally filters for contributors with long-term stewardship orientation.
Explicit evaluation and feedback	Progression decisions are formal and documented, conducted by higher-tier contributors using predefined benchmarks. Contributors receive constructive feedback on strengths and areas for improvement, with clear guidance on what is needed for advancement.
Recognition and incentive alignment	Milestone recognition at each tier. Financial incentives (if any) are tied to sustained contribution and progression, not to one-time deliverables. This structural choice is essential for filtering out speculative participants.

Anti-Sybil Design for Contributor Programs

Web3's permissionless nature and token-based incentives create sybil risk in contributor programs. OMF recommends the following design patterns:

Verified contribution history requirements	Pathway entry requires demonstrated prior contribution to open source (not necessarily within the ecosystem). This filters out participants with no genuine development background.

Staged access with minimal early financial incentive	Early pathway stages provide recognition and learning resources but minimal or no financial compensation. Contributors motivated by token speculation self-select out when early work is unpaid or modestly compensated maintenance labor.
Reputation-weighted progression	Advancement criteria include not only contribution volume but peer endorsement, review quality, and community engagement metrics. Gaming volume metrics alone is insufficient for progression.
On-chain contribution attestation	Pathway milestones and tier achievements can be recorded as on-chain attestations (using standards like Ethereum Attestation Service or ecosystem-specific credential systems). These provide verifiable, portable proof of contributor status that is harder to fabricate than self-reported claims.

Values-Alignment Through Program Structure

Rather than screening for values through subjective assessment, OMF recommends designing pathway structure so that values alignment emerges naturally from participation:

Early tasks: documentation, triage, testing	Contributors whose first interactions with the project involve understanding and maintaining existing code develop a fundamentally different orientation than those who begin with feature development. This structural choice is the most effective values-alignment mechanism available.
Institutional knowledge capture	Pathway participants are expected to document what they learn. This reduces expertise barriers for future contributors while reinforcing the stewardship orientation — maintaining and improving shared knowledge is core maintenance work.
Community integration requirements	Progression requires demonstrated participation in community discussions, responsiveness to other contributors' questions, and collaborative behavior in code review. Lone-wolf contribution patterns, however technically skilled, are insufficient for maintainer advancement.

Seasonal Program Concept

Time-bounded onboarding cohorts (modeled on Google Summer of Code) provide structured entry points for new contributors while managing the operational overhead of mentorship. Ecosystems can run ecosystem-specific seasonal programs with defined application periods, mentorship assignments, stipend support, and contribution milestones. These programs complement the ongoing contribution ladder by providing concentrated onboarding periods that feed participants into the permanent pathway.

Implementation Examples

Multiple ecosystems have developed contributor pathway programs that implement portions of the OMF specification:

CNCF LFX Mentorship	The reference implementation. 25 mentees became project maintainers since 2020. Structured pairing, stipend support, and defined contribution milestones. Integrates with Google Summer of Code and Outreachy. Participants continue contributing after completion and progress to reviewer and approver roles across cloud-native infrastructure.
Polkadot Fellowship	A merit-based technical collective with explicit tiered membership (rank 0–9) and clear progression criteria. Demonstrates that formalized contributor advancement structures are viable in decentralized ecosystems. Directly informs OMF's tiered progression model.
Ethereum Protocol Fellowship	A structured program onboarding developers into core protocol development. Participants receive direct mentorship from core contributors and work on real protocol improvements. The closest Web3 equivalent to CNCF's mentorship model at the protocol layer.
Midnight Aliit Fellowship	A quest-based onboarding system where participants earn points through technical contributions — writing tutorials, submitting pull requests, building sample DApps. The inaugural cohort consists of 17 fellows from 11 countries, including open-source maintainers and ZK researchers. Demonstrates structured contributor qualification through graduated challenges.
Hedera DevRel and Partner Program	Developer Advocate support with direct mentorship pathways. The Hedera Developer Playground provides a browser-based tool for building and experimenting in seconds, reducing onboarding friction to near zero. Demonstrates that lowering the technical barrier to first contribution is a design choice, not an inherent constraint.
Cardano Contribution Ladder	A four-stage progression framework (New Contributor → Committer → Trusted Committer → Core Maintainer) with explicit responsibilities, quantifiable progression criteria, and structured evaluation at each transition. One example of formalizing contributor advancement for Web3 infrastructure.

Note: No single implementation covers all OMF contributor pathway requirements. The specification is the standard; implementations are reference examples that each address different dimensions of the challenge.

Appendix E: Portfolio Allocation Model

Supports: Section 12 (Funding Architecture) and Section 14 (Portfolio Stewardship)

This appendix operationalizes the portfolio discipline described in the main framework. It provides a scoring formula, a worked allocation example, an ROI proxy framework, and tiered accountability structures that ecosystems can adopt and calibrate to their specific contexts.

Risk-Priority Scoring Formula

OMF's portfolio allocation begins with a risk-priority score for each infrastructure dependency identified in the dependency audit. The formula produces a numeric priority that drives funding allocation:

Priority Score = (Dependency Centrality × W1) + (Inverse Bus Factor × W2) + (Maintainer Burnout Risk × W3) + (Security Incident History × W4)

Dependency Centrality (W1 = 0.30)	Normalized count of ecosystem projects depending on this component. Scored 0–100 based on the dependency audit. A library depended on by 80% of ecosystem projects scores near 100; one with 5% scores near 5.
Inverse Bus Factor (W2 = 0.25)	Inverse of the number of active maintainers with deep codebase knowledge. Bus factor of 1 = score 100; bus factor of 2 = score 50; bus factor of 5 = score 20. Projects with a single maintainer receive maximum risk scoring.
Maintainer Burnout Risk (W3 = 0.25)	Composite of self-reported availability, recent contribution trend (declining = higher score), time-in-role without relief, and CHAOSS contributor sustainability indicators. Scored 0–100 by the program operator based on survey data and contribution analytics.
Security Incident History (W4 = 0.20)	Frequency and severity of past security incidents weighted by recency. Unresolved critical vulnerabilities score 100. Clean security history with rapid response times scores near 0.

These weights are defaults. Ecosystems should calibrate weights to reflect their specific risk tolerances and strategic priorities — an ecosystem with recent security incidents may increase W4; one with severe maintainer attrition may increase W3. Weights must be published before each allocation cycle.

Worked Allocation Example

Consider a hypothetical OMF implementation with a $500K annual retainer budget and 20 scored dependencies:

The top-5 scored dependencies (highest risk) receive 60% of the retainer budget ($300K). The next 5 receive 25% ($125K). The remaining 10 receive 15% ($75K) or are identified as coverage gaps requiring different program types (bounties, incubation). This produces a distribution where the most critical infrastructure receives the most support — fundamentally different from a proposal-based system where the most articulate advocate receives the most funding.

The worked example demonstrates a key OMF insight: portfolio discipline means the dependency graph drives allocation, not the volume of proposals received. A critical library with no advocate receives funding because the centrality score demands it. A well-advocated project with low centrality receives less because the portfolio cannot justify the opportunity cost.

Budget Scaling Over Time

OMF programs should scale incrementally as operational confidence grows. Reference scaling trajectories from the comparative landscape:

STA trajectory	Began at €3.5M in 2022, scaling to €17M planned for 2025. Demonstrates that starting modestly and expanding based on validated outcomes reduces political risk and governance resistance.
GitHub Secure OSS Fund	$1.25M across 125 projects at approximately $10K each. Demonstrates that meaningful impact is possible at modest per-project funding levels for security-specific interventions.
Protocol Guild	Ongoing growth of contributor registry with four-year vesting periods. Demonstrates that continuity funding scales through expanding the contributor base rather than increasing per-contributor allocation.
OMF recommendation	Start with $280–570K in year one (per the First 12 Months roadmap in Section 15). Scale to $1.5–4M by year three if program outcomes validate the investment. Never exceed 10% of treasury for operational programs.

ROI Proxy Framework

Direct ROI measurement for infrastructure maintenance is inherently difficult — the primary value is the absence of failure. OMF recommends the following proxy metrics for governance accountability:

Cost of comparable security incident	Heartbleed (OpenSSL) cost estimates range from $500M to billions in remediation across the global economy. A single critical vulnerability in a high-centrality Web3 dependency could expose equivalent ecosystem value. The retainer cost for preventing such an incident is orders of magnitude lower.
Cost of maintainer replacement	Recruiting, onboarding, and ramping a replacement maintainer for a complex infrastructure project typically takes 6–12 months and costs 3–5x the annual retainer in lost productivity and knowledge gaps. Maintaining existing maintainers is structurally cheaper than replacing them.
Avoided downtime value	For Web3 ecosystems, downtime in critical infrastructure (RPC nodes, indexers, consensus libraries) directly impacts transaction throughput and ecosystem economic activity. Quantify the per-hour economic impact and multiply by avoided downtime hours attributable to funded maintenance.
Contributor pipeline value	Each maintainer produced through a contributor pathway represents accumulated ecosystem investment in training,

	mentorship, and institutional knowledge. CNCF’s 25 mentee-to-maintainer conversions represent substantial avoided recruitment cost for the cloud-native ecosystem.

Opportunity Cost Accounting

Every funding commitment displaces alternatives. OMF requires that governance decisions include explicit opportunity cost documentation: what is being funded, what is not being funded, and why. The portfolio allocation model makes this visible by showing the full ranked list of scored dependencies alongside the funding cutoff line. Governance participants can see not just the funded portfolio but the unfunded gap — and can make informed decisions about whether to increase the budget or accept the residual risk.

Tiered Accountability Structures

Applying uniform verification rigor regardless of funding level creates unnecessary overhead for small allocations and insufficient oversight for large ones. OMF recommends tiered accountability:

Micro-grant tier (<$25K)	Lightweight reporting: quarterly summary, contribution metrics from public repositories, self-assessment. Minimal administrative overhead to prevent the application process from exceeding the grant value.
Growth tier ($25K–$250K)	Structured reporting: quarterly outcome reports with specific KPI tracking, mid-point governance review, evidence-based renewal with documented metrics. Standard OMF evaluation cadence.
Infrastructure tier (>$250K)	Full accountability: quarterly outcome reports with auditable financial records, dedicated governance oversight, annual portfolio review with explicit sunset criteria, public transparency reports. Commensurate with the scale of ecosystem investment.

References: STA budget data from Interoperable Europe Portal, 2024; GitHub Fund from GitHub Blog, 2024; Protocol Guild from Protocol Guild Documentation, 2023–2025; Heartbleed cost estimates from multiple industry sources

Appendix F: Ecosystem Health Dashboard Specification

Supports: Section 16 (Measuring Ecosystem Health) and Section 13 (Operational Governance)

This appendix transforms the health indicators in Section 16 into a concrete dashboard specification with baseline targets, governance decision rules, and implementation guidance. The measurement framework is grounded in CHAOSS community standards and supplemented by OpenSSF security assessment methodologies.

CHAOSS as the Primary Measurement Standard

The CHAOSS OSS Project Viability model provides the most comprehensive available framework for ecosystem health measurement. It organizes metrics into categories covering compliance and security, community health, governance clarity, and contributor sustainability. OMF adopts this model as its measurement foundation, with the following adaptations for Web3 contexts:

On-chain metrics supplement	Web3 ecosystems have unique measurement advantages: on-chain activity provides objective, verifiable data on protocol usage and treasury disbursements. OMF recommends combining on-chain metrics with CHAOSS off-chain analysis for a complete picture.
Qualitative maintainer health	Off-chain maintainer wellbeing — burnout, succession planning, institutional knowledge retention — requires periodic survey-based measurement that on-chain data cannot capture. OMF recommends CHAOSS-aligned surveys published as public ecosystem health reports.

Recommended Baseline Targets

The following targets are starting recommendations that ecosystems calibrate to their specific contexts. They are not absolute standards — they are governance discussion triggers.

Maintainer attrition rate	Target: below 15% annually across funded projects. Measured as the percentage of funded maintainers who leave the program or reduce engagement below threshold during the measurement period. Rising attrition is the most reliable leading indicator of ecosystem infrastructure instability.
Security vulnerability response time	Target: median response time under 30 days for reported vulnerabilities in funded projects. Measured from report to patch availability. Critical vulnerabilities (CVSS 9.0+) should have a separate target of under 7 days.
Bus factor for critical projects	Target: bus factor of 2 or greater for all top-20 centrality projects. Any top-20 project with bus factor of 1 is classified as a portfolio emergency and triggers immediate resilience program activation.
Contributor-to-maintainer conversion	Target: above 5% conversion rate per contributor pathway cohort. Measured as the percentage of pathway participants who achieve trusted committer status or equivalent within 18 months of pathway completion.

Dependency freshness (Libyears)	Target: cumulative Libyears below ecosystem-defined threshold (calibrated during first dependency audit). Projects exceeding threshold are flagged for dependency update sprints.
Release frequency	Target: at least one release per quarter for all actively funded projects. Measured as time between releases. Projects with releases more than 6 months apart trigger review for maintenance adequacy.

Governance Decision Rules

Metrics without decision rules are reports, not governance tools. OMF requires that each baseline target be linked to an explicit governance action:

Automatic renewal trigger	Funded project meets all baseline targets for two consecutive quarters. Renewal recommendation proceeds to governance without requiring full re-evaluation. Reduces governance fatigue for well-performing programs.
Review trigger	Any baseline target missed for one quarter. Program operator initiates formal review with the maintainer, documents causes, and establishes improvement targets. Continued funding during review.
Escalation trigger	Any critical target (security response, bus factor) missed for one quarter, OR any non-critical target missed for two consecutive quarters. Governance body directly involved in review. Funding may be modified or suspended.
Portfolio rebalancing trigger	Dependency centrality increases by more than 50% between annual audits (indicating a project has become significantly more critical). Immediate coverage gap assessment regardless of current funding status.
Emergency trigger	Top-20 centrality project reports bus factor of 1 AND maintainer signals burnout or departure intent. Immediate resilience program activation: succession planning, knowledge documentation, contributor pathway acceleration for that project.

Dashboard Layout Specification

An OMF ecosystem health dashboard should display the following views. This is a functional specification, not a UI prescription — ecosystems implement using whatever tooling fits their infrastructure.

Portfolio coverage map	Visual mapping of all top-N centrality dependencies against current OMF program coverage. Shows which critical infrastructure is funded, which is unfunded, and which is in transition. The primary governance planning view.
Risk heat map	Two-dimensional plot: dependency centrality (x-axis) versus sustainability risk score (y-axis). High-centrality, high-risk projects appear in the upper-right quadrant — the highest-priority sustainment targets. Color-coded by current funding status.

Program effectiveness trends	Time-series charts showing: retainer-funded project maintenance activity, bounty completion rates, contributor pathway progression rates, and treasury deployment efficiency. Governance uses these to assess whether programs are producing outcomes.
Flagged alerts	Active alerts for any governance decision rule triggers (reviews, escalations, emergencies). Sorted by severity. Each alert links to the relevant metric data and recommended governance action.
Treasury deployment efficiency	Ratio of funded program output to treasury expenditure. Includes coordination cost breakdown (per the Cost of Coordination framework in Section 12). Enables governance to assess whether operational overhead is proportionate to outcomes.

Implementation Precedents

Multiple tooling and measurement approaches inform this specification:

CHAOSS / Bitergia / GrimoireLab	The primary analytics platform for CHAOSS metrics. GrimoireLab 2.0 provides historical data tracking, identity management, and extensible dashboards. Bitergia produces repo maturity reports used by multiple ecosystems. Directly applicable to OMF dashboard implementation.
OpenSSF Scorecard	Automated security assessment generating a score for open source projects based on security practices. Provides a standardized security hygiene baseline that OMF implementations can integrate as one dashboard input.
STA milestone evaluation	Delivery measurement pattern: milestone-based contracts with defined acceptance criteria and structured evaluation. Demonstrates accountability in public-sector funding that OMF adapts for delivery-oriented programs.
Protocol Guild contribution tracking	Time-weighted contribution tracking for continuity measurement. Demonstrates that ongoing availability can be measured and weighted objectively through on-chain registry data.

References: CHAOSS Community, chaoss.community; OpenSSF Scorecard, scorecard.dev; GrimoireLab, chaoss.github.io/grimoirelab; Bitergia, bitergia.com

Appendix G: Risk Mitigation Protocols

Supports: Section 17 (Risks, Precedents & The Road Ahead) and Section 6 (Institutional Legitimacy)

Section 17 identifies five primary risks: centralization, program capture, token volatility, governance fatigue, and scope creep. This appendix defines concrete mitigation protocols for each — moving from identified risks to operational safeguards that ecosystems can implement.

Operator Replacement Protocol

The replaceability principle is OMF's primary structural safeguard against centralization and program capture. For replaceability to be meaningful, the replacement process must be defined before it is needed:

Governance vote threshold	OMF recommends supermajority (e.g., two-thirds) for operator replacement under normal circumstances, ensuring stability. Simple majority threshold applies when the operator has failed to meet explicit performance criteria documented in the program charter. The threshold should be defined in the governance authorization that establishes the OMF program.
Mandatory transition period	Minimum 90-day transition from the governance decision to complete handoff. During this period, the outgoing operator must: transfer all program documentation, introduce the incoming operator to funded maintainers, complete in-flight disbursements, and provide access to all program management infrastructure.
Documentation handoff requirements	The outgoing operator must deliver: complete program records (all funding decisions, evaluation results, and correspondence), current dependency map and centrality scores, active program status for all funded projects, and financial reconciliation of all disbursements. Without this documentation, the incoming operator cannot maintain program continuity.
Escrow for in-flight funds	Program funds authorized but not yet disbursed are held in escrow (multisig or smart contract) during the transition period. Neither the outgoing nor incoming operator has unilateral access. Release requires governance confirmation that the handoff is complete.
Continuity of maintainer support	Funded maintainers must not experience interruption in support during operator transitions. Retainer payments continue from escrow during the transition period. The maintainer's relationship is with the program, not the operator.

Precedent: Traditional OSS foundations (Apache, Linux Foundation) handle leadership transitions through structured processes with defined handoff requirements. The STA model provides public accountability through parliamentary oversight. These precedents inform OMF's protocol design.

Governance Fatigue Mitigation

On-chain governance participants can reasonably resist ongoing program oversight requirements if every operational decision requires full community engagement. OMF addresses this through structured delegation:

Default-approve for compliant programs	Programs meeting all KPI thresholds for two consecutive quarters receive automatic renewal recommendation. Governance confirms with a simple ratification rather than full re-evaluation. This dramatically reduces the number of active decisions governance must process.
Escalation-only governance involvement	Routine operational decisions (individual retainer evaluations, bounty approvals within authorized parameters, pathway cohort management) are handled by the program operator within governance-authorized boundaries. Governance acts only on flagged exceptions: budget overruns, scope expansion requests, operator performance concerns, or metric-triggered escalations.
Off-chain committee structure	Operational detail is managed by dedicated committees (technical review, program oversight) that operate off-chain and report to governance on a defined cadence. This separation is validated by the STA's supervisory board model and by multiple Web3 governance committee implementations that handle operational coordination without requiring full community votes on every decision.
Quarterly summary reporting	Rather than per-decision approval, governance receives consolidated quarterly reports covering: portfolio status, program outcomes, funding utilization, and any flagged issues requiring governance action. Reduces information overload while maintaining accountability.

Scope Creep Protocol

Operational programs naturally tend to expand beyond their original mandate. OMF provides structural constraints:

Explicit authorization boundaries	Each OMF program charter must define: authorized activities, budget envelope, target project criteria, and explicit exclusions. The charter is a governance document — activities outside the charter are unauthorized regardless of their perceived value.
Expansion requires new governance proposal	Any activity, funding commitment, or target population outside the current charter requires a separate governance proposal. The program operator cannot self-authorize scope expansion. This creates a structural friction that prevents gradual drift.
Annual scope reconfirmation	As part of the annual portfolio review, each program's scope is explicitly reconfirmed by governance. The review asks: is the current scope still appropriate? Should it be expanded, contracted, or modified? This prevents the accumulation of legacy scope that no longer serves ecosystem needs.
Program sunset criteria	Every program charter must include explicit conditions under which the program should be terminated: metric thresholds that indicate the program is no longer effective, ecosystem changes that render

	the program unnecessary, or budget constraints that require portfolio rebalancing. Sunset criteria are defined at authorization, not discovered during crisis.

Token Volatility Mitigation

For Web3 implementations where treasury funding is denominated in native tokens, volatility risk can collapse the real-dollar value of committed funding between approval and disbursement:

Stable-asset conversion	Convert authorized program budgets to stablecoins or fiat equivalents at the point of governance approval, not at the point of disbursement. This insulates program operations from token price fluctuation after the funding decision is made.
Multi-source funding diversification	Draw from all three OMF funding categories (treasury, endowment, corporate pool) to reduce dependence on any single source. Treasury volatility is offset by endowment stability; endowment growth timelines are offset by immediate treasury availability.
Operating reserves	Maintain a defined operating reserve (OMF recommends 3–6 months of program costs) in stable assets. This buffer ensures program continuity during periods of extreme token volatility without requiring emergency governance action.
Disbursement timing strategies	For retainer programs, disburse in regular intervals (monthly or quarterly) rather than large lump sums. This reduces the impact of any single conversion event and provides natural checkpoints for program review.

Program Capture Prevention

Program capture occurs when specific maintainers, funders, or governance participants gain disproportionate influence over program decisions. OMF provides structural safeguards:

Transparent allocation criteria	Selection rubrics and evaluation criteria must be published before each allocation cycle. Retroactive criteria changes to accommodate specific applicants are a capture signal and grounds for governance review of the program operator.
Evaluation committee rotation	Members of evaluation committees should rotate on a defined schedule (OMF recommends annual rotation of at least one-third of committee seats) to prevent the accumulation of personal relationships that bias decisions.
Public decision documentation	All selection decisions, renewal decisions, and sunset decisions must be publicly documented with rationale. Any stakeholder should be able to trace the logic from published criteria to specific outcomes.
Separation of execution from oversight	The entity executing programs (operator) must be structurally separate from the entity providing oversight (governance body). An operator that also controls its own oversight has effectively

	captured its program. This separation is the architectural foundation of OMF’s legitimacy model.

Implementation examples of governance separation that inform these protocols include: STA’s SPRIND subsidiary model (operational independence within public accountability), CNCF’s TAC/operational staff separation (technical governance distinct from program management), Protocol Guild’s custody-free disbursement (no intermediary holds funds), and Cardano’s POSM three-entity model (Security Council, OSC, and OSO with explicitly separated execution, oversight, and administration functions) as one Web3 pilot of this separation principle.

References: Apache Software Foundation governance model, apache.org; Linux Foundation leadership transition practices; STA governance from Sovereign Tech Agency, 2026; CNCF governance from contribute.cncf.io; Protocol Guild documentation, 2023–2025