EXECUTIVE FIELD GUIDE

Web3 Without the Hype

Control, change authority, dependency and accountability in decentralised systems

A practical guide for deciding whether decentralisation is necessary, where control actually sits, and who remains answerable when decision rights leave the organisation.

Jörn Green

Green Bear Consulting

Revised working edition · August 2026

EXECUTIVE SUMMARY

Follow the Control, Not the Label

The decision this guide is designed to improve

A Web3 proposal often arrives as a technology choice: chain, wallet, token, smart contract or governance model. The executive decision is more specific. Which identity, asset, execution or decision right should leave the organisation? Who receives that right? What becomes materially better, and which new obligations follow?

This guide answers those questions through the Web3 Control Map. It traces seven layers: identity, assets, execution, upgrade authority, dependencies, governance and legal accountability. For each layer, the map records the declared model, the actor or mechanism that can exercise authority, and the route available when a decision must be challenged, reversed or compensated.

Three conclusions

Decentralisation is a property of a specific layer, not a verdict on the whole product. A network may distribute validation while upgrades, recovery, interfaces or treasury control remain concentrated. Concentration can be appropriate; it must be visible and consistent with the public claim.

Reduced control does not remove accountability. Users, partners and regulators may still expect an identifiable organisation to explain the design, disclose material authority and provide a credible response when the system creates harm.

A decentralised component earns its place only when its benefit exceeds the added cost of governance, dependency management, recovery and legal exposure. If a conventional architecture delivers the same benefit with clearer control and stronger recourse, use it.

How to use this guide

Apply the Control Map at investment review, architecture approval, before launch and whenever a material change moves authority or user exposure. Then run the Leadership Test. Proceed when the benefit is demonstrated and the control model is survivable. Redesign when the benefit is real but avoidable exposure remains. Leave Web3 out when the case depends on the label rather than a right that genuinely needs to move.

Evidence note. The recurring marketplace is hypothetical. The public cases and technical claims are sourced in Chapter 8. This guide is an executive operating aid, not legal, investment, accounting or security advice.

CHAPTER ONE

Beyond the Label

Follow control, not branding

A leadership team is asked to approve a Web3 initiative. The proposal says the product will be decentralised, users will own their assets and the community will govern the system. Those claims may all be defensible. They may also describe only a small part of the architecture.

Before approving the investment, ignore the label and map the control.

Who can change the code? Who holds the keys? Who can pause execution, recover access, move treasury assets or alter the rules? Which parts depend on organisations that have no contract with you and no obligation to protect your customers?

“Decentralised” does not answer those questions. A network can distribute transaction validation while leaving contract upgrades with a small multisignature group. Users may interact through wallets while a custodian still controls their assets. Governance may be public while a foundation, delegate group or interface operator retains the authority that matters in an emergency.

This is why Web3 is a poor category for executive decision-making. It covers different technologies, ownership models and governance arrangements that do not create one common operating condition. A blockchain, token or wallet may be part of the design, but none tells you where power sits across the system.

The useful unit of analysis is the right being moved.

A company considering Web3 may want to move custody from the platform to the user, execution from an internal service to shared infrastructure, or a decision from management to a defined voting process. Each move can create value. Each also changes who can intervene, what can be reversed and where accountability lands when something goes wrong.

The rest of this guide uses a seven-layer Control Map:

- **Identity:** who can act, authenticate and recover access.
- **Assets:** who can hold, transfer, freeze or restore value.
- **Execution:** which actions run automatically and who can interrupt them.
- **Upgrade authority:** who can change system behaviour, under which process and with what notice.
- **Dependencies:** which chains, bridges, oracles, interfaces and operators must continue to work.
- **Governance:** who can propose, vote, delegate, execute and use emergency powers.
- **Legal accountability:** which entity remains answerable to users, regulators and counterparties.

A system can distribute control at one layer and concentrate it at another. That is not automatically a defect. Emergency authority may be necessary. A recoverable account may be safer for a particular customer group than uncompromising self-custody. A centrally operated interface may make an otherwise complex protocol usable. The problem begins when the declared story and the executable authority are different—or when leadership has never examined the difference.

The same discipline applies to claims about ownership and trust. A wallet authorises actions with keys; it does not by itself prove legal identity or ownership. A token can provide access, transfer or voting functions; it does not automatically grant equity, intellectual-property rights or a share of revenue. A blockchain can make a submitted record difficult to alter later; it cannot establish that the original statement about a physical asset or off-chain event was true.

These systems do not remove trust. They redistribute it. Trust may move from a company database to contract code, validators, key holders, oracle operators, governance delegates or an interface provider. Some intermediaries disappear. Others become less visible and harder to hold to account.

That redistribution is worthwhile only when it solves a real problem.

There are good reasons to place a right or record outside one organisation: users may need to carry an asset between services; counterparties may need a shared execution layer without appointing one operator; a community may require transparent rules for a shared treasury; or a market may benefit from a common record that no participant can quietly rewrite.

There are also many situations where Web3 adds cost without creating a corresponding benefit. If one company must control the product, if transactions need routine reversal, if the data comes from a trusted internal source, or if customers expect a named provider to recover mistakes, a conventional architecture may be more honest and effective. Adding a token or blockchain does not improve a product merely by making its control model harder to see.

The first decision is therefore not which chain, wallet or token standard to use. It is whether any part of the problem requires a right to leave the organisation.

Name the right, its recipient and the material benefit. Then map the control that remains. If the team cannot do that, do not approve a Web3 architecture.

CHAPTER TWO

The Web3 Control Map

Map executable authority, not declared intention

> **The Control Map is not a decentralisation score. It is a record of who can act, which powers they hold and what remedy exists when those powers fail or are abused.**

Consider a hypothetical proposal from a game publisher. Players would buy digital items, hold them through a wallet, trade them across a shared marketplace and vote on changes to transaction fees. The items would run on a public network. The pitch describes the system as player-owned and community-governed.

That description is not yet false. It is simply incomplete. The wallet may be recoverable by a service provider. The publisher may control the marketplace interface, item utility and contract upgrades. A token vote may recommend a fee change while a company-held multisignature account executes it. The public network may settle transactions even though the product still depends on one hosted gateway.

For each layer, record three things: the declared model, the authority that can actually execute a change and the available recovery or challenge path. Do not accept ‘the community’, ‘the protocol’ or ‘the company’ as sufficient answers. Name the contract, role, key threshold, legal entity or decision process.

Identity

Start with the account model. Who can create an account, authenticate an action, block access or recover a user? A wallet address does not prove a legal identity, and a wallet connection does not prove self-custody. Smart accounts can define their own validation logic; embedded wallets, custodians and recovery providers create other control arrangements.

In the hypothetical product, players sign in with email and receive an embedded wallet. That may be good product design. The map should still show whether the player, publisher or wallet provider can reset access, replace credentials or prevent transactions. ‘User-owned’ means little until that authority is explicit.

Assets

Next, separate control of the token from control of its value. Who can mint, transfer, freeze, burn or restore it? Does the player control transfer from the wallet? Can an administrator blacklist an address? Can the publisher change scarcity or withdraw the item from the game? What legal or contractual rights accompany the token?

A player may control the token while the publisher controls every reason to want it. If another game does not recognise the item, portability is technical rather than useful. The Control Map records both powers instead of collapsing them into the word ownership.

Execution

Identify what the system executes without discretionary approval: transfer, settlement, fee allocation, access or voting. Then identify who can interrupt that execution. A contract may run on a public network while the only practical route to it is a publisher-operated interface. Automated execution can coexist with concentrated access.

The same distinction applies to external facts. If an item changes when a tournament result is reported, the contract needs an input mechanism. The ledger can show which value was submitted and what followed; it cannot by itself establish that the reported result was correct.

Upgrade authority

Record every pause, upgrade and migration mechanism. Some contracts cannot be altered. Others use proxies, administrative roles or governance-controlled timelocks. A multisignature arrangement can reduce single-person control without making authority broad or public. A timelock can give users notice, but only if the proposal, delay, executor and cancellation rights are understood.

For the game marketplace, the relevant evidence is not a promise that upgrades will be rare. It is the current signer set, signature threshold, scope of the upgrade, delay before execution and action available to a user who disagrees.

Dependencies

List the network, scaling layer, bridge, wallet provider, gateway, data store, marketplace and any external data source that must continue to work. For each dependency, name its operator or governing body, failure mode, change authority, fallback and contractual recourse. Open infrastructure is not the same as infrastructure without an operator, and a protocol is not automatically outside human control.

A dependency the company cannot control may still be acceptable. It must be visible, bounded and designed around. If the hosted gateway fails, can users reach the contracts another way? If a bridge is paused, are assets trapped? If upstream governance changes fees or rules, who decides whether the product migrates?

Governance

Follow a decision from proposal to execution. Who may propose? Who may vote or delegate? What constitutes approval? Who executes the result? Who can override it in an emergency? Participation is not the same as authority. A public vote may be advisory, binding, delayed by a timelock or subject to a separate multisignature action.

If players vote on marketplace fees but the publisher can ignore the result, describe the vote as consultation. If the vote controls an executable contract, show the concentration of voting power and

the emergency path. The map should state what the mechanism does, not what the organisation hopes the word governance will imply.

Legal accountability

Finally, name the entity that offers the product, markets the token, operates the interface, holds customer data, provides recovery and answers complaints. Decentralised execution does not determine which law applies or remove the need for a responsible counterparty. In the European Union, MiCA establishes requirements for covered crypto-asset offers, admissions to trading and crypto-asset services; consumer, data, contract and other financial rules may also apply. The exact assessment belongs with qualified counsel, but the operating model cannot leave the accountable entity blank.

Read the whole map

The hypothetical product now looks more precise. Players may control transfers. The public network may execute settlement. Yet account recovery, item utility, the interface, upgrades and legal accountability may remain concentrated. Community input may exist without binding execution. None of this automatically disqualifies the design. It does disqualify the unqualified claim that the system is player-owned and community-governed.

The map produces one of three decisions. Proceed when the chosen distribution of authority creates a material benefit and the remaining concentration is disclosed and controlled. Redesign when the benefit is credible but recovery, dependencies, governance or upgrade power create avoidable exposure. Decline when the promised benefit depends on a decentralisation claim that the executable architecture does not support.

The map is complete when every material power points to a named actor or mechanism, every external dependency has a fallback or accepted failure boundary, and every user exposure has an accountable response. Approve that specific arrangement of authority—not a percentage, a label or an aspiration.

Source notes

Account validation and smart accounts: ERC-4337 — Account Abstraction Using Alt Mempool

Administrative roles and delayed execution: OpenZeppelin Contracts — Access Control and TimelockController

Off-chain inputs and oracle limitations: Ethereum.org — Oracles

EU regulatory scope and application: Regulation (EU) 2023/1114 on markets in crypto-assets

CHAPTER THREE

Five Operating Conditions

What the architecture requires the organisation to handle

The Control Map shows where authority sits. It does not yet show whether the organisation can operate the arrangement it has designed. Five conditions determine whether the hypothetical game marketplace is manageable after launch: change authority, custody and recovery, external dependency, public evidence, and governance with legal accountability.

The publisher from Chapter 2 has a credible reason to continue. Letting players transfer an item outside one game account may create useful portability and a market that the publisher does not solely administer. But the proposal still carries an unqualified promise: player-owned, community-governed and protected by code. The operating review tests what the company must be able to do when that promise meets a defect, a lost account, a failed dependency, an incident or a disputed decision.

1. Finality, upgradeability and change authority

Suppose the marketplace contract allocates transaction fees incorrectly. The team discovers the defect after assets have already moved. Calling the contract immutable does not resolve the situation. It only tells the team that one particular route to correction may be unavailable.

Several forms of change may coexist. A confirmed network transaction can be final while the application issues a refund or compensating transfer. Deployed code may remain at one address while a proxy points users to new logic. A contract may include a pause function, a migration path or an administrator able to alter a parameter. An interface can stop presenting an action even when users can still call the contract directly.

Each option changes the risk rather than removing it. A contract with no upgrade path can preserve a rule and preserve its defect. A broadly upgradeable contract can fix the defect and allow whoever controls the upgrade role to change the bargain after users have committed assets. A pause can contain harm and become an unexamined power to block legitimate use.

Before release, the publisher needs a change-authority record for every critical component: current owner, administrative roles, signer threshold, upgrade mechanism, pause authority, migration route, notice period and maximum credible blast radius. The record must also state what users can do before a change takes effect. A timelock creates meaningful notice only when the proposal is visible, the delay is long enough to act and users have a realistic exit or challenge path.

The approval question is concrete: can the organisation correct an unacceptable failure without granting an undisclosed actor the power to rewrite the system at will? If the answer is no, the architecture needs another release path, another control mechanism or a narrower promise.

2. Custody, recovery and user exposure

The marketplace uses an embedded wallet because most players will not manage seed phrases. The account appears inside the game, login begins with email and recovery feels familiar. That product choice may remove a major adoption barrier. It also decides who can restore access, replace credentials, block a transaction and respond when the recovery process is attacked.

A wallet connection does not reveal the custody model. An externally owned account controlled by one private key, a programmable smart account, a custodial account, an embedded wallet and a multisignature account create different obligations. Recovery may be impossible, delegated to guardians, administered by a provider or governed by contract logic. Multisignature can reduce dependence on one key while concentrating authority in a small signer group.

The design choice should follow the user and the loss scenario. If the player holds the only key, the product must explain signing risk and permanent loss before value enters the account. If the publisher or a provider can recover access, the company inherits a security process, an identity-verification problem and a remedy obligation. If recovery is programmable, the team must test who can invoke it, how providers are changed, what delay applies and how a false recovery is challenged.

Asset control also needs to be separated from asset value. A player may control transfer of the token while the publisher controls scarcity, item utility and whether the game recognises it. Support material cannot call the whole arrangement self-custodial or player-owned if the practical value still depends on company decisions that the player cannot contest.

Product, security, customer support and legal counsel therefore need one agreed account model before launch. It should state who can act in normal use, who can act during recovery, what evidence they require, what can be lost and which entity answers the complaint. Those are product requirements, not implementation details.

3. External protocols and dependency without recourse

The product depends on more than its own contracts. The chain must continue to settle transactions. The embedded-wallet service must authenticate users. A gateway must expose network data. A bridge may carry assets into another environment. An oracle may report an off-chain result that changes item behaviour. The marketplace interface and indexing service must turn raw activity into something a player can understand.

Conventional software already relies on open-source code and infrastructure outside direct control. The sharper distinction here is between dependency with contractual recourse and dependency without it. A commercial supplier can fail despite an agreement. A protocol can have maintainers, administrators and emergency procedures despite being described as decentralised. The operating model needs to identify which relationship exists for each component.

Bridges and oracles make the issue visible. A bridge adds contracts, validators or counterparties whose failure can interrupt transfer or expose assets. An oracle makes external information usable on chain, but the contract will execute against the submitted value whether or not the underlying report was complete. Recording the input does not verify the tournament result, the identity of the reporter or the integrity of the off-chain process that produced it.

The publisher needs a dependency register with six fields: component; operator or governing body; failure mode; change authority; fallback or migration route; and user exposure. 'No fallback' can be an accepted answer if the exposure is understood and the launch decision explicitly accepts it. An empty field is not acceptance.

For the marketplace, the review should test what happens if the wallet provider blocks recovery, the gateway disappears, the bridge pauses or upstream governance changes fees. The company may not control those events. It still controls whether customers enter the product without a bounded response.

4. Public evidence and incident response

An incident begins with two clocks. On-chain activity may be visible immediately; verified understanding develops more slowly. Observers can see a transfer, contract call or asset movement and form a plausible account before the publisher knows whether the cause was defective code, a compromised device, a false oracle input, stolen credentials or authorised behaviour with an unexpected effect.

The ledger provides evidence about recorded activity. It may not reveal legal identity, intent, internal approval, off-chain compromise or the complete economic impact. Public evidence changes incident tempo; it does not remove the need for investigation. A fast statement that confuses a transaction trace with a root cause can produce a second failure when later evidence contradicts it.

Before launch, the publisher should assign an incident commander, technical authority, communications owner, legal escalation and treasury or asset authority. The team also needs predefined conditions for pausing an interface, invoking a contract control, notifying users, preserving evidence and offering recovery or compensation. If different organisations control the wallet, bridge and contracts, the escalation path must identify how those organisations are reached and what they are actually able to do.

An incident update should separate four categories: what has been observed; what remains unverified; what has been contained; and what remedy is available now. That structure is more credible than either silence or premature certainty because it lets users distinguish evidence from interpretation.

5. Governance, power and legal accountability

The fee vote now reaches the point of execution. Players may submit proposals and vote with tokens, but the result enters a timelock and a designated executor sends the final transaction. An emergency group can pause the marketplace. The publisher controls the game logic that gives the items utility. Governance is therefore not one mechanism; it is the whole path from proposal to consequence.

The design must state who may propose, who may vote or delegate, how voting power is distributed, what constitutes approval, who executes and who can intervene. A transparent vote can remain concentrated. A formally approved proposal can remain non-binding. A timelock can create review time or merely delay an outcome that no user can change. Emergency authority can protect assets and bypass the public process.

None of those arrangements is automatically illegitimate. The defect appears when the public description hides the binding authority. If the publisher can disregard the fee vote, call it consultation. If the vote binds a contract, disclose the executor, delay, concentration and emergency exception. If a foundation or multisignature group can alter the process, name it.

Legal accountability follows a separate analysis. In the European Union, MiCA establishes requirements for covered crypto-asset offers, admissions to trading and crypto-asset services. Other consumer, data, contract and financial rules may apply according to the product and actors involved. A decentralised component does not determine the legal classification or make the responsible entity disappear. The exact assessment belongs with qualified counsel; the operating design still needs to identify the entity that markets the product, operates the interface, provides custody or recovery and answers complaints.

After this review, the marketplace can still proceed. The claim must change. Players control defined transfer rights; a public network executes settlement; the publisher and named providers retain recovery, interface, upgrade and emergency powers; a specified governance process controls fee decisions; and a legal entity remains accountable for the service. That statement is less dramatic and much more useful.

The operating review is complete only when each condition has an owner, evidence, an intervention path and a user remedy. If one condition depends on an unnamed actor or an untested assumption, the Control Map is describing an aspiration rather than an operable product.

Source notes

Contract change patterns: Ethereum.org — Upgrading smart contracts

Administrative roles: OpenZeppelin Contracts — Access Control

Delayed governance execution: OpenZeppelin Contracts — TimelockController

Account-recovery interface proposal: ERC-7947 — Account Abstraction Recovery Interface

External data boundaries: Ethereum.org — Oracles

Cross-chain dependency risk: Ethereum.org — Bridges

EU regulatory scope: Regulation (EU) 2023/1114 on markets in crypto-assets

CHAPTER FOUR

Operating at Two Speeds

Match the governance to the cost and authority of change

A Web3 product does not have one release cadence. It contains decisions with different owners, reversal paths and consequences. The operating mistake is to classify every interface change as fast and every on-chain change as slow. The useful distinction is whether a change is bounded and recoverable, or whether it alters rights, assets or the conditions under which other people depend on the system.

Return to the game marketplace. Changing the colour of an item card may be routine. Changing the wallet screen that asks a player to sign a transfer is not: a small interface edit can change consent and loss exposure. Updating a contract parameter may be routine if the range is constrained, the owner is known and the effect is reversible. Migrating item contracts may require notice, external coordination and a user decision even when the code change itself is technically simple.

Classify the change before scheduling it

Every material component should be classified through five fields. First, authority: who may approve and execute the change? Second, mechanism: configuration, deployment, upgrade, governance proposal or migration. Third, lead time: how long is required for testing, notice and external coordination? Fourth, reversibility: can the previous state be restored, compensated or only replaced? Fifth, blast radius: which users, assets, integrations and legal obligations can be affected?

Those fields place work into one of two operating lanes. The routine lane covers changes with a named owner, short lead time, limited exposure and a tested reversal path. The governed lane covers changes that move assets, alter rights, expand administrative power, depend on external approval, require user migration or create consequences that cannot be credibly undone.

The lanes are not labels attached permanently to technologies. A user-interface change can enter the governed lane when it affects custody, disclosure or consent. An on-chain parameter can enter the routine lane when authority is narrowly delegated, allowable values are bounded, monitoring is in place and reversal is defined. Classification follows the decision, not the location of the code.

The routine lane

Routine changes should move quickly because their uncertainty and exposure are controlled. The release record needs an owner, test evidence, monitoring signal and rollback condition. Approval can remain with the product or engineering owner inside pre-agreed limits. Examples might include copy

that does not alter a signing decision, an indexer improvement, a non-custodial interface fix or a contract parameter adjusted within an approved range.

Speed still depends on observability. If the team cannot detect whether the change harmed transaction completion, displayed the wrong asset state or produced unexpected calls, it does not have a fast lane. It has an unobserved lane.

The governed lane

Governed changes need a wider evidence package because they can alter the bargain. The record should identify the affected control layers, authority path, threat model, dependency impact, user notice, migration or exit route, incident owner and remedy if the decision fails. Approval may require security, product, operations and legal review, followed by a multisignature action, governance vote or timelock.

For the marketplace, changing mint authority, replacing the wallet provider, upgrading transfer logic, moving assets across a bridge or redefining the fee vote belongs here. The relevant question is not whether the change can be deployed today. It is whether every party who bears the consequence has the evidence, notice and recourse the operating model promises.

Escalation between lanes

A routine change must move to the governed lane when any of five triggers appears: the owner or change mechanism is different from the approved record; the reversal path has not been tested; the blast radius includes assets or user rights; an upstream protocol or provider must coordinate; or the team cannot state the remedy if the change fails. The release calendar should not override those triggers.

The reverse movement also matters. A repeatedly reviewed parameter change can become routine after the authority is narrowed, acceptable ranges are encoded, monitoring is reliable and rollback has been demonstrated. Good governance does not make every decision slow. It invests once in boundaries that allow safe decisions to move faster.

The portfolio decision

At each planning review, the leadership team should see which changes sit in each lane, who owns them and what evidence is missing. A backlog organised only by features conceals the decisions that can change rights or expose assets. A separate governance queue conceals the fact that some governance decisions are ordinary operational work. One portfolio, classified by change conditions, keeps both visible.

The two-speed model is working when routine work moves without executive theatre and high-consequence work cannot pass through a routine release by accident. If the team cannot explain why a change belongs in its lane, it has not earned the cadence it is using.

Source basis: Ethereum.org — Upgrading smart contracts; OpenZeppelin — Access Control; OpenZeppelin — TimelockController

CHAPTER FIVE

From Product to Institution—Sometimes

Recognise when shared infrastructure creates durable obligations

The two-speed model classifies changes by authority and consequence. A harder question follows: what exactly is being governed? A feature remains part of a product when one organisation can define it, change it and withdraw it within an ordinary customer relationship. A system becomes institution-like when several parties rely on shared rules, assets or decision rights that are expected to survive any one release or management team.

The threshold is durable dependence

A blockchain does not make a product an institution. Neither does a token, a vote or a public treasury. The threshold is crossed when the system starts settling questions that participants cannot resolve through a normal supplier decision: who belongs, what they own, which rules bind them, how collective assets are used, how disputes are handled and how the arrangement continues when one operator leaves.

Six conditions make the shift visible. Membership creates standing rather than mere access. Shared assets persist beyond a single transaction. Rules apply across organisations or user groups. A decision process can change those rules. An execution path turns decisions into consequences. A continuity mechanism covers dispute, emergency, succession and exit. A system need not satisfy all six perfectly, but the more of them it carries, the less credible it is to manage governance as a feature owned by one product team.

This is also where the word 'community' becomes dangerous. A community can describe an audience, a customer segment, a group of contributors or a body with formal rights. Those are different arrangements. If members can discuss but cannot propose, decide or execute, the system offers participation without authority. That may be entirely appropriate. It should be named accurately.

The marketplace crosses the line in stages

Return to the game marketplace. If the publisher sells items that work only inside its own game, sets every rule and can close the service under published terms, the arrangement remains a product. Putting the item record on a public chain changes the architecture, but it does not by itself create a shared institution. The publisher still controls the utility that gives the item most of its value.

The position changes when several games recognise the same asset, outside marketplaces depend on the transfer standard, fees accumulate in a common treasury and token holders receive binding rights over selected parameters. The rules now coordinate publishers, players, wallet providers, market operators and contract administrators. A unilateral change can strand integrations or alter rights that another party has treated as durable.

At that point the publisher must decide whether it actually wants to create shared infrastructure. If the answer is no, it should narrow the promise: retain clear control, use consultation where useful and offer contractual commitments that customers can understand. If the answer is yes, governance needs an institutional specification covering membership, authority, execution, emergency powers, appeal, funding and succession. The token contract is only one part of that specification.

Governance must cover the whole decision path

A voting screen can make governance look complete while leaving the consequential work elsewhere. The operating record must show who may place a proposal, who determines eligibility, how votes or delegations are counted, whether the result is binding, who executes it, which delay applies, who can intervene in an emergency and where a participant can challenge an error. It must also show who pays for maintenance after the founding company changes priorities.

Each gap creates an undeclared centre of power. If an administrator selects which proposals reach a vote, agenda control sits with the administrator. If a multisignature group can refuse execution, the vote is conditional. If a foundation can replace the voting mechanism, constitutional authority sits with the foundation. None of those choices is disqualifying. The problem appears when the public claim assigns power to the community while the executable path assigns it somewhere else.

Digital governance is larger than Web3

Decidim and Pol.is are useful counterexamples because they demonstrate governance capabilities without depending on a blockchain. Decidim is open-source participatory-democracy infrastructure built in Ruby on Rails. Organisations can configure proposals, assemblies, consultations, participatory budgets and accountability processes. Pol.is gathers statements and agree, disagree or pass responses, then uses the resulting opinion matrix to identify groups and areas of broad agreement.

Neither platform is evidence that Web3 governance works; both are evidence that deliberation, agenda design, transparency and participation are separate design problems. A chain can record a vote or execute an approved transfer. It cannot determine whether the eligible population was legitimate, whether minority positions were heard, whether participants understood the proposal or whether the decision process deserved trust.

This creates a practical selection rule. Use on-chain execution when the decision must control shared digital assets or rules without relying entirely on one operator. Use conventional governance tools when the need is participation, deliberation or accountability and a named institution can execute the result. Combine them only when the hand-off between discussion, decision and execution is explicit.

The institutional decision

Before launch, leadership should state which future it is approving. In the product version, the company retains authority and offers bounded user rights, consultation and contractual recourse. In the shared-infrastructure version, multiple parties receive durable standing and the company accepts limits on unilateral change. An ambiguous middle—community language with revocable rights and hidden execution power—creates the cost of an institution without a defensible constitution.

The approval gate is therefore simple: identify the rule, asset or decision right that must persist beyond the founding operator, then show the membership, authority, execution, remedy and succession arrangements that will carry it. If no such right needs to persist, keep the system a product.

Source notes

Adjacent digital governance: Decidim: general description; Decidim: participatory processes; Computational Democracy Project: Pol.is

CHAPTER SIX

Three Cases Under the Control Map

Test the framework against failure, functioning governance and concealed concentration

A framework earns its place by explaining cases that do not all point in the same direction. The three cases below are not a ranking of projects and they do not prove that Web3 succeeds or fails as a category. Each tests a different claim: that code can contain governance, that distributed decision-making can produce a bounded outcome, and that a decentralised architecture can conceal a concentrated path to control.

The same questions apply in every case. Where did identity, assets, execution, change authority, dependencies, governance and legal accountability actually sit? What happened when the stated model met a defect or a real allocation decision? What can the case support—and what would be an overclaim?

1. The DAO: code did not contain the constitutional question

The DAO opened its token sale on 30 April 2016 and received about 12 million ETH before the sale closed on 28 May. Its contracts were intended to pool capital and let token holders participate in funding decisions. On 17 June, an attacker exploited a recursive-call vulnerability in the DAO contract and began moving ether into a child DAO. The protocol behaved according to deployed code while producing an outcome most participants regarded as unacceptable.

The Control Map exposes the mismatch. Execution sat in the DAO contracts, but the practical power to remedy the event did not. The affected ether could not be recovered through the DAO's ordinary process. Developers proposed changes at the Ethereum network layer; miners and node operators had to decide which software to run; exchanges and users had to decide which chain and asset they would recognise. The incident therefore moved from application code into social and protocol governance.

Ethereum implemented the DAO hard fork at block 1,920,000 on 20 July 2016. Ethereum.org records that more than 85 per cent of votes cast supported the fork, while some miners rejected it and continued the unforked chain as Ethereum Classic. The fork restored a withdrawal path for DAO token holders; it did not erase the disagreement. The system had discovered a constitutional authority that the product description had not fully specified: participants could coordinate a protocol change, and dissenters could preserve the previous rules by sustaining another chain.

Legal accountability remained outside the code. In July 2017, the US Securities and Exchange Commission concluded that the DAO tokens, under the facts it examined, were securities. The SEC's narrower lesson is useful here: automating functions through smart contracts did not remove the activity from existing law. The case does not prove that every serious incident will be forked. It shows that a claim of fixed execution is incomplete unless the wider system names who can coordinate exceptional change and what happens when others refuse.

2. Optimism Retro Funding 5: governance worked because the mandate was bounded

Retro Funding 5 was designed to reward contributions to the OP Stack and Ethereum. The round reviewed 149 applications and approved 79 after an application-review and appeal process. Citizens then made allocation decisions within a defined voting design. The final round distributed 8 million OP in October 2024 across Ethereum core contributions, OP Stack research and development, and OP Stack tooling. The unit matters: this was an OP allocation, not a dollar claim that can be repeated without a valuation date.

This is a functioning governance case because the process produced a traceable decision and transfer, not because every layer was decentralised. The Foundation set the round scope, application rules, minimum and maximum allocation range, KYC requirements and grant-delivery mechanism. Reviewers decided which applications met the rules. Citizens voted on the round size, category shares and project allocations. Grants were streamed after results and KYC approval. Different actors held agenda, eligibility, allocation and execution authority.

The round also tested expertise rather than treating all participation as interchangeable. Twenty-eight guest voters selected for proximity to OP Stack work participated alongside Citizens. Optimism's 2024 citizenship review reported differences in allocation patterns: voters with greater familiarity found technical impact easier to assess, while non-experts tended toward flatter allocations. That finding does not establish one objectively correct distribution. It shows that voter selection and information design affected the output and were treated as variables to examine.

The limitation is as important as the result. Retro Funding 5 did not demonstrate community control over the whole Optimism system. It demonstrated that a mixed governance arrangement could allocate a specified pool through declared roles and produce an inspectable outcome. Calling it bounded does not diminish the achievement. It identifies why it could work: the mandate, eligible population, decision sequence, asset pool and delivery path were narrower than the ambition to 'govern the protocol.'

3. Ronin: nine validators concealed a five-signature path

The 2022 Ronin bridge incident shows why counting nodes is not the same as mapping control. The bridge used nine validators and required five signatures for a withdrawal. Four validators were controlled by Sky Mavis. A temporary arrangement had also allowlisted Sky Mavis to obtain signatures through an Axie DAO validator during a period of high load. The operational need ended, but the permission was not revoked.

After compromising Sky Mavis systems, the attacker controlled the four Sky Mavis validator keys and used the stale allowlist path to obtain the fifth signature. Two fraudulent withdrawals on 23 March removed 173,600 ETH and 25.5 million USDC. The breach was discovered six days later after a user reported being unable to withdraw 5,000 ETH. Public transactions existed throughout that period; the organisation lacked the monitoring needed to interpret them as an incident.

The Control Map would have treated the four company-controlled validators, the fifth-signature delegation and the withdrawal threshold as one effective authority path. The network had several validator identities, but bridge execution could be captured through one organisation's systems plus an obsolete permission. The architecture distributed labels more widely than it distributed the ability to move assets.

Ronin's response also shows that accountability returned to identifiable organisations. Sky Mavis and Axie Infinity committed to cover users, and Ronin reported in June 2022 that user funds were again backed one to one. The reopened bridge added audits, higher signature requirements for large withdrawals, human review, a seven-day delay for the largest tier, a daily withdrawal limit and a circuit breaker. Those controls are forms of governance and intervention, not evidence that code made intervention unnecessary.

This case describes the bridge architecture and incident in March 2022, not Ronin's current decentralisation state. Its lesson is narrower: a threshold number is meaningless without common-control analysis, permission lifecycle and monitoring. A board reviewing a bridge should ask how many independent failures are required to authorise loss—not how many validator names appear on a diagram.

What the cases change

The DAO shows that exceptional change can migrate from contract execution to protocol and social authority. Optimism shows that distributed governance can produce a real outcome when its mandate and execution path are explicit, even while a foundation retains important powers. Ronin shows that apparently distributed components can collapse into one control path through shared systems and stale delegation.

None of the three supports a category verdict. Together they support a review discipline: map executable authority, observe who acts when the system is stressed, and describe the arrangement at the narrowest level the evidence can defend. The next chapter turns that discipline into an approval test.

Source notes

The DAO: Ethereum Foundation incident update; Ethereum fork history; SEC investigation report
Optimism Retro Funding 5: Round details; Application review; Citizenship learnings 2024
Ronin: Bridge reopening report; US Treasury DeFi risk assessment

CHAPTER SEVEN

The Leadership Test

Decide whether to proceed, redesign or leave Web3 out

A proposal can arrive with audited contracts, a respected chain, a governance forum and a detailed launch plan, yet still be unfit for approval. Those materials describe components. Leadership must decide whether a right should leave the organisation, whether the resulting control model is acceptable and who will answer when the system produces an outcome that customers, counterparties or regulators reject. The test below turns that decision into six evidence gates.

Run the test before an investment commitment, at architecture approval, before launch and after any change that moves authority or user exposure. Each answer needs an owner and a current record. A dashboard that labels a gate green is not evidence; the underlying keys, rules, tests, contracts and operating arrangements are. An unresolved gap that can authorise asset movement, remove recovery or shift accountability blocks approval. Other gaps become dated launch conditions rather than assumptions hidden inside the plan.

1. Prove necessity

Name the user or business problem that requires distributed ownership, execution or decision rights. Then identify the smallest decentralised component capable of delivering the benefit. Portability across independent services, settlement without appointing one operator or durable rights in shared infrastructure may justify the added complexity. Fashion, fundraising language and a preference for token economics do not.

Evidence required: a specific right, its intended holder, the conventional alternative and the material advantage created by moving that right outside the company.

2. Map effective control

Complete the Control Map with names rather than categories. Record the people, entities, keys, contracts, thresholds and processes that govern identity, assets, execution, upgrades, dependencies, governance and legal accountability. Treat several roles administered through the same organisation or system as one control path. A disclosed centre of authority may be appropriate. An undisclosed one invalidates the claim.

Evidence required: current role assignments, signing thresholds, delegation and recovery rules, upgrade and pause mechanisms, infrastructure operators and the legal entities behind them.

3. Trace one credible failure

Choose the failure with the largest plausible user consequence and trace it from cause to detection, containment, recovery and compensation. Include shared systems and stale permissions, not only the component that appears on the architecture diagram. State what can be reversed, what can only be compensated and what cannot be repaired.

Evidence required: a tested incident path, monitoring ownership, emergency authority, communication triggers, funding for recovery and a decision on who may impose loss.

4. Follow governance into execution

A vote is not a governance system. Show who can submit a proposal, decide eligibility, set the agenda, vote or delegate, count the result, execute it and intervene in an emergency. Report concentration and expected participation. Identify decisions that remain off-chain and any administrator who can delay, filter or refuse the result.

Evidence required: the full proposal-to-execution path, including quorum, time delay, veto, appeal, emergency and succession arrangements.

5. Make user exposure and accountability explicit

Describe what a user can lose, misunderstand or be unable to reverse. Specify custody and recovery options, the support route, dispute handling and any compensation boundary. Then name the entity responsible for the product in every market served. A decentralised execution path does not provide a legal conclusion, and a disclaimer cannot substitute for determining which obligations apply.

Evidence required: plain-language risk disclosure, recovery design, terms and complaints route, jurisdictional analysis and a named executive owner for unresolved exposure.

6. Record the decision

Proceed when the decentralised element creates a demonstrated benefit, the effective control model matches the public claim, the main failure path is survivable and accountability is accepted. Record the remaining risks and the conditions that would trigger a pause or review.

Redesign when the benefit is real but control is concentrated accidentally, recovery is inadequate, a dependency lacks a fallback, governance cannot execute reliably or users carry exposure they cannot reasonably understand. Narrow the promise or the decentralised scope before approving capital or launch.

Do not use Web3 when a conventional system can deliver the same benefit with clearer control, stronger recourse and lower operating cost, or when the proposal depends on calling a centrally controlled arrangement decentralised. The approval record must name the right that leaves the organisation, who receives it, which burdens are accepted and why that trade remains worthwhile. If it cannot, the decision is not ready.

CHAPTER EIGHT

Sources and Evidence Boundaries

What this guide can support—and what it cannot

This guide separates observed facts, interpretations and operating recommendations. A source can establish that a transaction occurred, a rule was published or an organisation reported an outcome. It rarely proves the whole claim made about a system. Each case is therefore described at the narrowest level its evidence supports.

How the examples are labelled

The game marketplace used across Chapters 2 to 5 is hypothetical. It exists to keep one product decision visible while control, change, failure and governance are examined; it is not a disguised client case. The DAO, Optimism Retro Funding 5 and the Ronin bridge are public cases based on the sources listed below. No anonymised advisory experience or composite client story is presented as evidence in this edition.

A future case should be labelled public and sourced, verified first-person experience, anonymised advisory case, composite illustration or hypothetical. Anonymisation protects identity; it does not relax the need to state the decision, mechanism, date range and evidence limit.

What different records establish

An on-chain record can show that a specified address submitted a transaction and that the network recorded a state change. It does not by itself verify the person's identity, the truth of an off-chain input, legal ownership, informed consent or intent. Technical documentation describes an available mechanism; it does not prove that a deployed product configured, secured or operated that mechanism correctly. Governance posts and project reports are useful primary accounts of rules and reported outcomes, but self-reporting is identified as such and checked against independent or regulatory material where the claim warrants it.

Dates, quantities and legal scope

Volatile quantities are reported in their original unit with an event date. A token allocation is not converted into dollars unless the valuation date, price source and calculation are stated. Case descriptions refer to the architecture and governance arrangements in the period examined, not necessarily their current state. Source links were checked for this working edition on 7 August 2026.

MiCA is used to establish the limited point that covered crypto-asset activities in the European Union operate within a legal framework; it is not used to classify every token or product discussed here. Applicability also depends on the activity, parties, jurisdiction and facts. This publication is an executive operating guide, not legal, investment, accounting or security advice. A Control Map does not replace legal analysis, smart-contract review, threat modelling or an independent audit.

The claim this guide makes

The Control Map is a due-diligence method for locating authority and exposing mismatches between a public claim and an executable system. It does not produce a universal decentralisation score or predict every failure. Its test is narrower: can leadership identify where consequential rights sit, what can change, which dependencies can fail and who remains answerable? Decisions made after the source-check date should revalidate the technical, governance and legal material that affects them.

Source notes

Ethereum.org — Smart contracts — Architecture and persistence of deployed contract code.

Ethereum.org — Upgrading smart contracts — Proxy, migration and upgrade-authority patterns.

Ethereum.org — Oracles — The boundary between on-chain execution and off-chain inputs.

Ethereum.org — Bridges — Cross-chain designs, dependencies and risk categories.

ERC-4337 — Account Abstraction Using Alt Mempool — Programmable smart-account architecture.

ERC-7947 — Account Abstraction Recovery Interface — A proposed standard interface for account recovery.

ERC-2981 — NFT Royalty Standard — Royalty information and the voluntary-payment boundary.

OpenZeppelin Contracts — Access Control — Administrative roles, multisignature ownership and delayed execution.

Regulation (EU) 2023/1114 — Markets in crypto-assets — Primary EU legal text used for the limited regulatory framing in this guide.

Ethereum Foundation — Critical Update Re: DAO Vulnerability — Contemporaneous account of the exploit and response options.

Ethereum.org — Ethereum fork history — DAO fork date, block and continuation of Ethereum Classic.

US SEC — Report of Investigation: The DAO — Token-sale chronology and the SEC's fact-specific legal analysis.

Optimism Governance — Retro Funding 5 round details — Round scope, allocation design and grant-delivery conditions.

Optimism Governance — Retro Funding 5 application review — Application counts, review process and appeals.

Optimism Governance — Citizenship learnings 2024 — Reported allocation outcome and guest-voter findings.

Ronin Network — The Ronin Bridge Is Open — Project report on losses, reimbursement and reopening controls.

US Treasury — DeFi Illicit Finance Risk Assessment — Government assessment of the Ronin compromise path and bridge risks.

Decidim documentation — General description — Adjacent evidence about digital participation infrastructure; not a Web3 case.

Computational Democracy Project — Pol.is — Adjacent evidence about large-scale deliberation and opinion mapping.

ABOUT THE AUTHOR

Jörn Green

Technology and operations leader · Green Bear Consulting

Jörn Green is a technology and operations executive with more than 25 years of experience across software, gaming, telecommunications and IoT. He has led product and technology organisations in Europe and Asia, with responsibility for strategy, delivery, commercial outcomes and organisational scale.

At Sony, he led mobile software and firmware-over-the-air operations reaching approximately 100 million devices, built an organisation in China from one person to around 200, and took an IoT business from concept to its first automotive customer and more than EUR 2 million in annual revenue. At Ubisoft, he scaled Ubisoft Connect from approximately 40 to 240 people while its user base grew from 40 million to 60 million. At Play'n GO, he led a 300-350-person games organisation with more than EUR 35 million in P&L responsibility and increased annual output from 52 to 64 games, a 22% improvement.

Through Green Bear Consulting, Jörn works with technology leadership teams when growth, complexity or uncertainty has outpaced the way the organisation operates. His work focuses on decision rights, operating discipline, commercial evidence and leadership structures that convert strategy into coordinated execution.

Web3 Without the Hype applies that operating perspective to systems where control and decision rights may leave the organisation while accountability remains.

Read Jörn's work at jorngreen.online

Connect with Jörn Green on LinkedIn