What a Game Can Be

https://gerolds.github.io/textbook/textbook/posts/what-a-game-can-be/

Contents

What a Game Can Be

1. TL;DR

- A game needs more than theme. It needs a situation where action can be repeated, judged, varied, and intensified.
- Six gates test whether source material contains a game: context → pressure → loop class → primitive → escalation → proof.
- The core primitive must be obvious, readable, interesting, observable, repeatable, and progressive.
- A working primitive is not enough. The designer must be able to say what the player is here to do and why that’s satisfying. Mechanics without intent produce toys, not games.
- A concept that passes all six gates *speaks for itself*. It aligns the team, resists drift, and makes bad decisions obvious even to someone who wasn’t in the room when the concept was written.
- If one person cannot prove the primitive in two weeks, the concept is not ready for production. It may not be a game at all.
- Not every interesting subject survives this process. That is the point.

---

2. Why this essay exists

The companion essays assume a game concept already exists. *Wide Appeal* *(https://gerolds.github.io/posts/wide-appeal/)* evaluates a finished work. *Making the Thing* *(https://gerolds.github.io/posts/making-the-thing/)* builds around a governing commitment. *Prototyping the Loop* *(https://gerolds.github.io/posts/prototyping-the-loop/)* finds the invariant. *Finding the Commitment* *(https://gerolds.github.io/posts/finding-the-commitment/)* explains why sticking matters.

This essay covers the prior step: testing whether a subject — a historical period, a profession, a conflict, a system — contains a viable game before committing a team to it. Six gates, applied in sequence, force the subject to prove it contains playable pressure, a classifiable loop, and a core action one person can build in two

weeks. The process must be able to reject. A method that accepts every subject tests nothing.

---

3. **The six gates**

The process runs through six gates. Each gate asks a question the source material must answer before you advance. Failing a gate is not failure; it is information. The material may be valuable elsewhere — atmosphere, narrative, simulation detail, worldbuilding — but not as a game.

> **Context → Pressure → Loop Class → Primitive → Escalation → Proof**

Six checkpoints. Each one tests the ground before you put weight on it.

3.1. **The thread that runs alongside: intent**

The gates test whether a subject contains a *working game*. They do not test whether anyone would want to play it.

Intent is the designer's answer to: *What's the fantasy? What does the player want to be doing when they sit down?*

Not theme. Not setting. Not "the player will feel tense." The specific thing the player is pretending to do, and what makes that pretending fun. A game about Age of Exploration navigation is not "about exploration." The fantasy might be *you're the captain, the charts are probably wrong, and every morning you have to pick a heading anyway*. A player hearing that sentence either leans in or doesn't. That's intent doing its job.

Intent also protects the concept. A clear intent sentence is a pollution detector. When someone proposes a feature or a pivot, you hold it against the intent. If it fits, build it. If it contradicts, the contradiction is visible to everyone in the room, not just the lead designer.

State your intent at Gate 4. Test whether strangers feel it at Gate 6. Intent is not a gate. It is the reason the gates matter.

4. Gate 1: Context — understand the system before you design for it

Every subject is a system. A historical period has actors with competing incentives. A profession has constraints, tools, and failure modes. A location has geography, resources, and pressure points. A conflict has sides, escalation logic, and resolution conditions.

Your job at this gate is research, not design. You are not looking for "the game" yet. You are looking for how the system works: who acts, what they want, what constrains them, what breaks, what repeats.

What to extract:

- **Actors.** Who participates? What are their roles?
- **Incentives.** What do actors want? What are they optimizing for?
- **Constraints.** What limits action? Time, resources, rules, geography, information?
- **Feedback.** How does the system respond to action? How do actors learn what worked?
- **Failure modes.** What goes wrong? What are the characteristic ways the system breaks?
- **Recurrence.** What cycles? What repeats with variation?

Example: You are interested in the Age of Exploration. Actors: crowns, merchants, navigators, indigenous populations. Incentives: trade routes, territorial claim, religious mission, survival. Constraints: wind, provisions, disease, navigation technology, crew morale. Feedback: return on investment, maps, colonial reports. Failure modes: mutiny, shipwreck, scurvy, misidentified landfall. Recurrence: the voyage cycle — outfit, depart, navigate, encounter, return or die.

At this stage you should feel the system's logic. If you don't, you haven't researched enough. The game won't be *about* the subject; it will be *built from its pressure*.

Gate check: Can you describe the system's actors, incentives, constraints, and recurrence without referencing games? If the material produces only a static description (a place, an era, a mood) it may not contain a game. It might be setting, not structure.

---

5. Gate 2: Pressure — locate the recurring tension

Pressure is the engine of play: the thing that forces decisions, creates stakes, and makes repetition non-trivial. Not every interesting system produces it.

Pressure is *recurring tension between what an actor wants and what the system allows*. It must repeat naturally, vary without authorial intervention, and produce situations where different actions lead to different outcomes.

Types of pressure:

- **Scarcity.** Not enough of something: time, resources, space, attention, trust.
- **Uncertainty.** Incomplete information that forces commitment before resolution.
- **Opposition.** Another actor (or the system itself) working against you.
- **Deterioration.** Things getting worse unless you act.
- **Incompatibility.** Multiple goals that cannot all be satisfied.
- **Exposure.** Risk that increases with action.

The Age of Exploration example produces several: scarcity of provisions and crew health, uncertainty of navigation, opposition from weather and hostile encounters, deterioration of morale, incompatibility between speed and safety, exposure to disease and storm the further you venture.

What makes pressure *playable*:

The pressure must produce *decisions*: moments where the player chooses between meaningfully different actions. Pressure that

produces only one rational response is scripting: the player performs the correct action and waits for the next prompt. Pressure that produces no legible options is noise — the player acts at random and attributes the outcome to luck.

Both break the feedback loop. Scripting teaches nothing because the outcome was predetermined. Noise teaches nothing because the outcome can't be traced to the choice. A game whose loop doesn't teach can't retain a player past the first session. For an indie studio, retention *is* viability.

Ask: *What does the pressure force someone to decide?* If the answer is a genuine tradeoff (speed vs. safety, risk vs. certainty, investment vs. conservation) you have playable pressure.

Gate check: Can you name a recurring pressure that forces a tradeoff, with two or more reasonable responses? If the subject produces only observation ("this is fascinating") but not decision ("what would you do?"), it has texture but no pressure. Texture is not structure.

---

6. **Gate 3: Loop class — match the pressure to a known structure**

Pressure does not become play until it resolves into a repeatable cycle of action, feedback, and adaptation. That cycle is a loop.

Loops cluster into recognizable families. Not by logical necessity, but by practical constraint. A loop must cycle fast enough for the player to learn from repetition, produce legible feedback, and sustain variation without changing identity. These three requirements — speed, legibility, stability — filter the space of possible cycles hard. What survives resembles what else survives.

Classifying pressure into a loop class does two things. It stops you from pretending any subject can become any game. And it forces the subject to resolve into a recognizable structure rather than a thematic costume draped over nothing.

6.2. **The loop catalogue**

The ten types below are not exhaustive. They are the families this essay recognizes based on what ships and what an indie team can build around. A loop that doesn’t fit any of them is not necessarily wrong, but it is unproven, and unproven loops are expensive bets. If your pressure resolves into something genuinely novel, treat that as a risk factor, not a badge of honor, and test it harder.

Optimization loop The player improves efficiency under constraint. *Cycle:* allocate → execute → measure outcome → refine. *Works for:* logistics, economics, scheduling, city systems, survival planning, agriculture.

Pursuit / evasion loop The player closes distance or avoids capture through prediction and adaptation. *Cycle:* track → position → commit → escape or intercept → reset under higher stakes. *Works for:* hunting, stealth, policing, predator-prey dynamics, military maneuver, espionage.

Territory / control loop The player expands, defends, and restructures space. *Cycle:* claim → fortify → pressure weak points → absorb losses → expand again. *Works for:* war, politics, ecology, gang rivalry, infrastructure, colonization.

Risk management loop The player acts under uncertainty and mounting downside. *Cycle:* assess risk → commit resources → absorb variance → stabilize or spiral. *Works for:* medicine, finance, disaster response, expeditions, crisis governance.

Detection / inference loop The player converts incomplete signals into conclusions. *Cycle:* observe → hypothesize → test → revise model → act on interpretation. *Works for:* investigation, diagnosis, espionage, archaeology, intelligence analysis.

Negotiation / leverage loop The player reads interests and applies pressure through exchange. *Cycle:* infer motive → offer or threaten → observe reaction → reposition. *Works for:* diplomacy, trade, labor conflict, court politics, hostage situations.

Timing / execution loop The player succeeds by rhythm, precision, and sequencing. *Cycle:* read window → commit action → recover → chain into next window. *Works for:* combat, sports, music, machinery, ritualized performance.

Engine-building loop The player assembles interacting systems that produce compounding returns. *Cycle:* invest → unlock interaction → accelerate output → reinvest under new constraints. *Works for:* industry, colony growth, deckbuilding, research systems, automation.

Endurance / attrition loop The player survives deterioration over time. *Cycle:* conserve → endure → patch losses → choose sacrifice → continue under worse conditions. *Works for:* sieges, illness, harsh travel, collapsing societies, horror, survival.

Social deduction / trust loop The player acts through incomplete loyalty and manipulated perception. *Cycle:* signal → read signals → test trust → expose or deceive → social reset. *Works for:* conspiracy, hidden roles, court intrigue, resistance movements.

6.3. **Using the catalogue**

Most subjects map to one primary loop and one or two secondary loops. The Age of Exploration resolves primarily as a **risk management loop** (the voyage as a sequence of bets against deterioration) with secondary **detection/inference** (navigation, landfall identification) and **endurance/attrition** (crew survival under scarcity).

The primary loop is the one you build around. Secondary loops add depth but don’t drive the minute-to-minute experience. This is not a taste preference. It is a scope constraint. Every loop needs its own primitive, feedback system, escalation path, and proof. An indie team that builds around two co-equal loops is building two games and shipping neither. Pick the loop with the strongest pressure. Let the others flavor it.

If you can’t identify a primary loop, the subject may be too diffuse, or you haven’t found the strongest pressure yet.

Gate check: Does the pressure map to a recognizable loop class? Can you write the cycle in four to six steps? If the interesting thing is static, observational, or resolvable in a single action, it is probably not a game.

7. Gate 4: Primitive — extract the core action

The loop tells you the shape of the cycle. The *primitive* is the smallest unit of meaningful action inside it — the thing the player actually *does* every few seconds. The identity molecule of the game. The thing people mime when they describe it.

The primitive is not a genre label (“combat,” “building,” “trading”). It is a concrete action with testable properties. A good primitive is:

Obvious. A new player sees what to do within seconds. Not every nuance, but the basic action. If it requires explanation before the player can attempt it, it isn’t obvious enough.

Why this is necessary: A primitive that requires explanation before the player can *attempt* it inserts a barrier between the player and the loop. The loop is where learning happens. Everything before it is cost the player pays before receiving any value. An obvious primitive lets the player enter the loop immediately and learn by doing. The nuance can be infinite; the entry point cannot be. Indie games lack the brand trust that buys patience for onboarding. A stranger on a store page gives you seconds, not minutes.

Readable. The outcome of each attempt is legible. The player can tell whether they did well or poorly. Feedback is immediate or nearly so.

Why this is necessary: Readability turns repetition into learning. If the player can connect their action to its outcome — “I chose this heading and ended up here” — the loop teaches. If the connection is opaque, the player attributes results to luck or system whim. An unreadable primitive cannot be progressive, because the player has no basis for improving. It cannot retain either, because a game that feels random punishes investment.

Interesting. More than one reasonable way to perform it. The primitive admits variation, style, and choice. One correct answer makes it a reflex, not a decision.

Why this is necessary: A primitive with one correct answer has a skill ceiling of “learn the answer.” Once found, the loop generates no new information. Multiple viable approaches with different risk/reward profiles are what make each cycle produce a different

situation. They are also what make players *disagree* about the best approach, which is the structural basis of discussion and community. An indie game lives or dies by whether players talk about it. They talk about decisions, not reflexes.

Observable. Someone watching can see the primitive and understand what's happening. A primitive that only makes sense to the person holding the controller is hard to transmit.

Why this is necessary: This is about how games spread. An indie studio without a marketing budget depends on the game *showing itself* — streams, screenshots, word of mouth. All require the primitive to be legible to someone who isn't playing. If the interesting part of your game is an internal mental model with no visible output, the game may be excellent and no one will ever know. Large studios can buy awareness. Indie studios need the game to generate its own.

Repeatable. It can be performed hundreds of times without becoming rote. Each instance differs enough that the player is making a fresh decision, not replaying a memory.

Why this is necessary: The primitive *is* the game. If it exhausts itself after twenty repetitions, the game is twenty repetitions long, and no amount of content or narrative wrapper changes that. Repeatability requires that the primitive's inputs vary — through randomness, player history, system state, or opponent behavior — so each cycle is a fresh decision. This is also a production argument: a repeatable primitive generates play from its own structure. An exhaustible one requires constant content investment, and content is the most expensive thing an indie team produces per hour of play it creates.

Progressive. There is a difference between performing it adequately and performing it well. Skill, knowledge, or judgment produce better outcomes. Low floor, high ceiling.

7.4. **Deriving the primitive**

Look inside your loop cycle. Find the step where the player makes the most consequential choice in the shortest time. That is your candidate.

In a **risk management loop** for Age of Exploration: the candidate is *the navigation commitment* — choose a heading on incomplete information and live with the consequences until the next observation. Obvious (pick a direction), readable (you find out whether you're on course), interesting (multiple headings are plausible), observable (the ship moves), repeatable (every day at sea is a new reading), progressive (each leg of the voyage produces weather patterns, current behavior, and landmark sightings that are useless in isolation but compound into a mental model of the region; an experienced navigator doesn't guess less, but guesses better because prior commitments generated signals that inform the next one; the game's tension shifts from "which way?" to "how much of what I think I know should I trust?").

In a **detection/inference loop** for archaeology: the candidate is *the dig decision* — choose where to excavate based on surface clues, survey data, and theory. Obvious (pick a site), readable (what you uncover tells you whether you chose well), interesting (clues support multiple hypotheses), observable (the ground opens), repeatable (every excavation is a fresh test), progressive (every dig produces fragments — pottery shards, soil layers, structural remnants — that mean nothing alone but accumulate into a web of signals about what the site contains and where; the player forms hypotheses, commits scarce excavation resources to test them, and each outcome refines or breaks the model; skill is not "read stratigraphy faster" but "hold three competing theories in mind while deciding which one to risk disproving next").

In an **endurance/attrition loop** for a besieged city: the candidate is *the rationing choice* — decide who gets what from a shrinking pool. Obvious (distribute supplies), readable (people survive or don't), interesting (every allocation trades one survival against another), observable (the city changes), repeatable (every day is a new distribution), progressive (each rationing round reveals how different groups respond to shortage — who hoards, who shares, who breaks, who becomes dangerous; an experienced player reads these social signals and allocates not just for today's survival but to prevent tomorrow's crisis; the game's depth is not "manage numbers better" but "learn to predict second-order consequences of scarcity on a population you increasingly understand").

7.5. **The primitive test**

Write the primitive in one sentence:

> *The player [verb] [object] under [constraint], and [feedback].*

Examples:

- “The player commits a heading under uncertain winds, and the next landfall reveals whether the reading was right.”
- “The player allocates rations under scarcity, and dawn reveals who survived and who didn’t.”
- “The player places a tile under spatial pressure, and the board state reveals new threats and opportunities.”

If you cannot write this sentence, you do not yet have a primitive. Go back to the loop and look for the moment of highest decision density.

7.6. **The intent statement**

Write the intent in one sentence:

> *This game is about [what the player is doing] — [why that’s satisfying], expressed through [the primitive].*

This sentence does two jobs. It tells the team what to build. And it makes bad ideas visible: any proposed feature that contradicts it produces an obvious conflict.

Keep it concrete. If a developer on the team can’t read it and immediately start thinking about features, it’s too abstract. “Feel the weight of moral tradeoffs” is a theme lecture. “Pick which crew members eat today knowing someone won’t” is a design prompt.

Examples:

- “This game is about *running a trade route where you load cargo before you know what the market wants* — the fun is gambling on instinct, expressed through buy-and-ship commitments under uncertain prices.”
- “This game is about *keeping a village alive through winter by splitting rations that are never enough* — the fun is tough calls

with visible stakes, expressed through daily allocation under shrinking reserves.”

- “This game is about *rebuilding a dig site from broken pieces where every placement locks in an interpretation* — the fun is slow-burn detective work, expressed through tile placement with permanent consequences.”

The intent statement is not marketing copy. It is the shortest pitch that lets the team build in the same direction. Someone proposes adding real-time combat to the dig-site game. Everyone in the room can see the problem without the lead designer explaining it.

Gate check: Can you name the primitive, write the one-sentence formula, and write the intent statement? If the primitive is vague or fails multiple property tests, the concept is still a theme looking for a verb. If the primitive works but the intent is generic (“the player will have fun”), abstract (“experience the tension of choice”), or something a player can’t picture before touching the game, the concept has mechanics but no pull.

---

8. Gate 5: Escalation — prove the primitive survives repetition

A primitive that works once is a toy. A primitive that works a hundred times is a game.

Can the primitive sustain repeated play by introducing new pressure, new context, new wrinkles without losing its identity? If the primitive has to become a *different* primitive to stay interesting, it was not deep enough.

Three tests:

8.7. Variation test

Can the same primitive produce meaningfully different situations?

The primitive’s *inputs* must change even though the *action* stays the same. If the inputs are static, the player solves it once and the loop dies. Variation must come from something outside the player’s

control (randomness, opponent behavior, system state) that feeds *through* the primitive, forcing a different decision each cycle.

Test: list the sources of variation in your loop. For each one, ask: does this change what the player *decides* when performing the primitive, or does it only change what happens after? Variation that alters the decision is structural. Variation that only alters the outcome is decoration, and decoration is a slot machine. If new situations require actions that bypass the primitive, the primitive is not carrying the game.

8.8. **Depth test**

Does skill matter more over time, not less?

A primitive with depth rewards investment. The hundredth hour should feel different from the first — not because the game added systems, but because the player got better. They read situations faster, commit more efficiently, see consequences further ahead.

Depth that comes from system expansion — new unlocks, new mechanics, new modes — requires content to sustain. Each expansion must be designed, built, and balanced. For a large studio with a content pipeline, this is expensive but feasible. For an indie team, it is a treadmill with a hard ceiling. Depth that comes from the *player* improving at a stable primitive costs nothing to sustain. Chess has not shipped a content update in fifteen hundred years. The depth is in the player, not the game, and that is the only kind of depth an indie team can afford.

Ask: *What does an expert do differently from a beginner, using the same primitive?* If the answer is “nothing, they’re just faster,” the primitive lacks depth. If the answer involves judgment, prediction, risk calibration, or creative adaptation, it has room to grow.

8.9. **Stakes test**

Can the consequences of the primitive escalate without the primitive itself becoming absurd?

Early in a game, a navigation error might cost a day. Later, it might cost the expedition. The primitive (commit a heading) stays the same; the stakes compound.

Unhealthy escalation changes the primitive: navigation becomes fleet management becomes politics becomes a different game. If the only way to raise stakes is to add systems that dilute the primitive, the concept has a ceiling.

Ask: *Can I double the stakes three times without changing what the player does?* If yes, the primitive scales. If no, you will hit a wall in the midgame, and the typical response — adding features — is how coherence dies.

Gate check: Does the primitive pass all three escalation tests? Can you describe what play looks like in hour one, hour ten, and hour fifty, with the primitive still central? If the primitive exhausts itself or requires systemic escape hatches to stay interesting, the concept is not yet viable.

---

9. Gate 6: Proof — one person, two weeks, playable

A concept that survives Gates 1–5 on paper is a hypothesis. Proof costs time, and time is the only resource that matters.

The rule: **one person builds a playable demonstration of the core primitive in two weeks.** Not a demo. Not a vertical slice. A playable proof that the primitive works — obvious, readable, interesting, observable, repeatable, progressive — without borrowing meaning from anything outside itself.

This is where the concept either speaks for itself or doesn't. The *proof slice* *(https://gerolds.github.io/posts/prototyping-the-loop/)* is the smallest build that lets a stranger say, unprompted, "Oh, I get it. This is a game where I ___." No pitch deck. No context briefing. The concept communicates or it fails.

9.10. What the proof must demonstrate

- **The primitive is playable.** A person can sit down and perform it without instruction.
- **Feedback is legible.** The player can tell whether they did well or poorly.

- **Repetition is tolerable.** The player performs it ten or more times without checking out.
- **Improvement is visible.** The player gets better and can feel it.
- **Intent is legible.** The player's language after a session echoes the intent. If the game is about loading cargo before you know the market, listen for "I was sure silk would pay off but I should have hedged with tea." The concept is communicating without you in the room.
- **No crutches.** No progression systems, no narrative wrapper, no production art disguising a shallow loop.

9.11. **What the proof does not need**

- Final art or sound.
- Multiple systems or metagame.
- Content variety beyond what the primitive generates.
- Narrative context.
- More than one loop.

9.12. **Why two weeks**

Why not one week? Why not four?

One week is too short to test *repetition*. You can build a primitive in a week, but you cannot build enough variation around it to know whether the tenth play feels different from the first. A one-week proof tests whether the primitive *exists*. It does not test whether the primitive *sustains*.

Four weeks is too long because it hides dependency. A primitive that needs four weeks of surrounding context to land is borrowing meaning from systems that won't exist at launch. It also costs twice as much to fail, and failure is the expected outcome. Most concepts don't survive proof. The process depends on that being cheap.

Two weeks sits at the boundary: long enough to build a loop that cycles multiple times with variation, short enough that the primitive must carry the experience without crutches. If the primitive cannot stand alone in this window, it either depends on

other systems (not a primitive) or is too complex to teach itself (not legible in the market).

The two-week window also sets the rhythm of exploration. A month can test two concepts. A month of four-week proofs tests one. Committing before comparing is how studios lock into concepts that feel inevitable only because nothing else was tried.

The two-week constraint also tests *legible promise* *(https://gerolds.github.io/posts/wide-appeal/)*. If the proof slice requires explanation, it fails. If it requires context the player doesn't have, it fails. The primitive must teach its own contract.

9.13. **After the proof**

If the proof works — strangers play it and the primitive lands — you have something worth committing to. Now the other essays apply: *Finding the Commitment* *(https://gerolds.github.io/posts/finding-the-commitment/)* for locking direction, *Making the Thing* *(https://gerolds.github.io/posts/making-the-thing/)* for the production sequence, *Prototyping the Loop* *(https://gerolds.github.io/posts/prototyping-the-loop/)* for deepening the invariant, *The Studio Primitive* *(https://gerolds.github.io/posts/studio-os/)* for the operational loop.

If the proof doesn't work, you have saved yourself months. Try a different primitive from the same loop, a different loop from the same pressure, a different pressure from the same context, or a different subject entirely.

Failure at the proof gate costs two weeks. Failure after twelve months of production costs everything.

Gate check: Did one person build it in two weeks? Did a stranger play it, describe the primitive unprompted, and echo the intent? Did they want to play again? If the primitive lands but the intent doesn't, you have a working toy. If both land, you have a concept that can survive production.

10. **The rejection criteria**

The six gates have a shadow: conditions under which a subject should be rejected as game material. These are not insults to the subject. They are respect for the medium’s demands.

A subject probably does not yield a strong game when:

- **No repeatable decision pressure.** The interesting thing happens once, or every situation resolves the same way.
- **No escalation headroom.** Past a natural ceiling, intensification feels forced or morally incoherent.
- **Primarily observational.** The pleasure is in witnessing, not acting. Powerful, but not a game.
- **One obvious action pattern.** The “right move” is always clear, so there is no decision and no loop.
- **Resists abstraction.** Formalization kills what is interesting about the subject. Some material is powerful precisely because it is ambiguous or irreducible.
- **Texture without loop.** A fascinating place, event, or profession gives worldbuilding and tone, yet fails to generate a playable cycle.
- **Mechanics without intent.** The loop works, the primitive passes every test, but the designer can’t say what the player is here to do. A concept like this has no immune system. Every producer, every meeting, every playtest will push it in a different direction, and nothing in the design pushes back.

Rejection is not permanent. A subject that fails today might succeed with a different lens, loop class, or primitive. But rejection-right-now is essential. Without it, the process is wishful thinking with steps.

---

11. **Applying the process: a worked example**

A compressed pass through all six gates using one subject.

Subject: The East India Company, 17th century — joint-stock trade expeditions to Asia.

11.14. **Gate 1: Context**

Actors: investors (risk capital), directors (route and cargo decisions), captains (voyage execution), crews (labor under coercion), local traders (supply and negotiation leverage), rival companies (competition for routes and ports). Incentives: return on investment, monopoly control, career advancement, survival. Constraints: monsoon seasons, cargo spoilage, disease, navigation limits, capital scarcity, political instability at ports. Feedback: profit or loss on return, reputation with investors, crew survival rates. Failure modes: shipwreck, mutiny, market collapse, diplomatic incident, cargo spoilage, piracy. Recurrence: the expedition cycle — raise capital, outfit, sail, negotiate, load cargo, return, settle accounts.

✓ The system has actors, incentives, constraints, and a natural cycle.

11.15. **Gate 2: Pressure**

Primary pressure: **incompatibility.** Every expedition forces tradeoffs between speed and safety, profit and crew welfare, diversification and focus, diplomatic caution and commercial aggression. Secondary pressures: scarcity (capital, provisions, favorable winds), uncertainty (market prices at destination, political conditions), deterioration (spoilage, morale, ship condition).

✓ The pressure produces genuine tradeoffs with multiple reasonable responses.

11.16. **Gate 3: Loop class**

Primary: **risk management loop.** Each expedition is a sequence of bets against compounding uncertainty. Assess conditions → commit capital and route → absorb variance (weather, markets, diplomacy) → return and settle accounts → reinvest or fold.

Secondary: **negotiation/leverage loop** (trade at port), **endurance/attrition loop** (crew and ship survival during voyage).

✓ The pressure maps cleanly to a risk management loop.

11.17. **Gate 4: Primitive**

The moment of highest decision density: **the cargo commitment at port.** Limited hold space, limited capital, uncertain home-market prices, perishable goods, and a window before monsoon closes. What to buy, how much, and when to leave.

> *The player commits cargo under uncertainty and hold constraint, and the return voyage reveals whether the bet was right.*

Obvious: choose what to load. Readable: profit or loss on arrival. Interesting: spices are high margin but perishable; textiles are stable but low yield; rare goods are speculative. Observable: the hold fills visually. Repeatable: every port, every season. Progressive (each expedition produces market intelligence that compounds across voyages into a mental model of trade patterns; an experienced trader reads signals a novice doesn't yet recognize; the skill ceiling is not "pick the expensive goods" but "hedge against three plausible futures using information you gathered by being wrong last season").

Intent: *This game is about loading cargo before monsoon season when you don't know what the market wants — the fun is gambling on your read of the situation, expressed through cargo commitments under uncertainty.*

✓ The primitive is concrete, passes all six tests, and has a stated intent. (Test: someone suggests adding ship-to-ship combat. The intent sentence makes the mismatch obvious.)

11.18. **Gate 5: Escalation**

Variation: different ports, seasons, political conditions, rival activity, and accumulated reputation generate unique situations each expedition. The primitive never changes; the context around it does.

Depth: a novice fills the hold with whatever is cheapest. An expert reads tea-price signals from the previous season, hedges perishables against textiles, times departure to catch favorable monsoon, and reserves hold space for opportunistic rare finds. Same action, different judgment.

Stakes: early expeditions risk modest capital. Later, the player's entire trading network, investor trust, and crew loyalty are on the line. The cargo commitment stays the same; its consequences multiply.

✓ The primitive survives escalation across all three tests.

11.19. **Gate 6: Proof**

One person, two weeks. Build a text-and-card prototype (digital or physical). Each round: draw port conditions, choose cargo from a constrained menu, commit departure timing, draw voyage events, resolve at home port. No progression systems. No narrative. Just the commitment and its consequences.

Can a stranger play it? The rules fit on one card. Does the primitive land? The player should feel the weight of the cargo decision within three rounds. Does the intent come through? Listen for "I *knew* I should have taken the textiles" or "I gambled on pepper and it paid off." Do they want to play again? If variation and stakes are calibrated, yes.

✓ Provable in scope.

---

12. **The process as a thinking tool**

The six gates are not a creativity method. They are survival equipment for a hostile environment.

The environment is hostile because the space of possible game subjects is enormous and almost all of it is empty. This is not pessimism; it is structure. A viable game requires a subject that produces recurring pressure, resolves into a cyclical loop, yields a testable primitive, survives escalation, and proves out in two weeks. Each requirement is independent and filters out a large portion of candidates. The survival rate is multiplicative, not additive — if each gate passes 30% of subjects, six gates pass under half a percent. "Interesting" clears one bar. "Viable" clears six.

For an indie team this is not academic. You have funding for one concept, maybe two. The market releases thousands of games a year. A concept that lacks *legible promise* *(https://gerolds.github.io/posts/wide-appeal/)* will never be seen. One that lacks structural depth will be seen and forgotten. The cost of building the wrong concept is not a bad quarter; it is the studio. Theme, setting, and atmosphere seduce because they feel like progress. They are not progress. They are the canopy that hides whether there is solid ground underneath.

The gates test the ground. Each one costs less than the next. Context is free. Pressure costs a conversation. Loop classification costs a whiteboard session. The primitive costs a few days of thought. Escalation costs a week of argument. Proof costs two weeks of building. If the subject fails at any point, you've lost the minimum possible time.

The process is deliberately linear because linearity prevents the most common failure in concept development: falling in love with a subject before proving it contains a game. It also produces a concept that communicates. A concept that passed all six gates can be written on a card, handed to a stranger, and understood. It tells the team what to build. It makes drift visible. When a producer suggests adding base-building to a game about cargo gambling, the intent statement makes the contradiction legible to everyone in the room.

That is what a good concept does. It doesn't need you to protect it. It protects itself.

A full pass — raw subject to playable proof — should take less than a month. If it takes longer, the subject is resisting, the designer is decorating instead of testing, or the primitive is too complex. All three tell you something useful.

Use this process before the others. Run a subject through the gates before you *find your commitment* *(https://gerolds.github.io/posts/finding-the-commitment/)*. Prove the primitive before you *prototype the loop* *(https://gerolds.github.io/posts/prototyping-the-loop/)*. Confirm a game exists before you ask whether it's *widely appealing* *(https://gerolds.github.io/posts/wide-appeal/)*.

Otherwise you are committing to a subject, not a game. Subjects, no matter how fascinating, do not ship.

---

13. **What this guide cannot do**

A field guide creates a dangerous feeling: the feeling of competence. You read it, you understand the vocabulary, you can answer every prompt. The gates become a checklist. The checklist produces a document. The document looks like design.

It isn’t.

The guide can structure your thinking. It cannot supply the thinking. “Describe the player” is easy to say and easy to answer. Marketing will hand you a demographic sentence in ten seconds. That sentence is worthless. The guide wants something harder: a description that tells the team *what this specific player finds satisfying about this specific loop*, stated plainly enough that it constrains production decisions. If the answer could apply to any game in the genre, it has no content. The prompt was answered but the question was not.

This is the pattern that kills. Every gate in this essay can be passed on paper by someone who follows the format without fighting the substance. The result is a concept document that reads well, survives a review meeting, and enters production. Three years later the project ships or doesn’t, and either way nobody can explain what went wrong because the documentation was clean. Stakeholders will frame the outcome as success. Teams will frame it as circumstances. The truth is simpler: the concept was dead from the start because nobody proved it was alive.

Rote advice in game development — “find the fun,” “prototype early,” “build a vertical slice” — fails for exactly this reason. The advice is correct. Following it produces activity that looks like progress. But no amount of correct process can substitute for the moment where a designer looks at their own concept and says *this doesn’t work*. That is not a gate you can schedule. It is the willingness to be wrong about something you care about.

This guide provokes that confrontation. It cannot guarantee it happens. If every gate passes on the first try, if the document contains no surprises, if the designer never felt uncomfortable, the guide was used as a template, not a tool. The concept is no more alive than what an LLM would produce, and for the same reason: nothing resisted during its creation.

The guide tests the ground. It does not walk it for you.

---

14. **Quick reference**

The six gates:

Gate	Question	Fail condition
1. Context	What are the actors, incentives, constraints, and cycles?	Static description with no systemic logic
2. Pressure	What recurring tension forces tradeoffs?	Interesting but observational; no decisions
3. Loop class	Which known loop type does the pressure resolve into?	Pressure that resists cyclical structure
4. Primitive	What concrete action is the player performing every few seconds?	Vague, genre-level, or fails the six-property test
5. Escalation	Does the primitive sustain variation, depth, and rising stakes?	Exhausts itself or requires escape systems
6. Proof	Can one person prove it in two weeks?	Requires explanation, crutches, or surrounding systems

The thread: Intent runs alongside all six gates. Stated at Gate 4, tested at Gate 6. A concept with clear intent defends itself against drift and meddling.

The primitive must be:

Property	Test
Obvious	New player sees what to do within seconds
Readable	Outcome is immediately legible
Interesting	Multiple reasonable approaches exist
Observable	A watcher can follow the action
Repeatable	Hundredth instance differs from the first
Progressive	Skill produces measurably better outcomes

The one-sentence formula:

> *The player [verb] [object] under [constraint], and [feedback].*

The intent statement:

> *This game is about [what the player is doing] — [why that's satisfying], expressed through [the primitive].*

The rejection criteria:

No repeatable decisions. No escalation headroom. Primarily observational. One obvious action. Resists abstraction. Texture without loop. Mechanics without intent.

---

15. **Appendix: a pathology of concepts that shouldn't have survived**

The six gates catch structural failures: no loop, no primitive, no escalation. But most bad concepts don't fail structurally. They fail politically. They enter the pipeline wrapped in language that sounds like design but isn't, and nobody in the room has the tools — or the authority — to call it out.

Each pathology below would not survive the gates if the gates were applied honestly. They survive because the gates were softened, skipped, or applied after the decision was already made.

15.20. **Evasive terminology**

The concept is described in words that sound specific but commit to nothing. “Emergent player-driven narrative.” “Systemic sandbox with deep social simulation.” “Meaningful choices with lasting consequences.” Every studio in the industry could paste these phrases into their pitch deck and nothing would change. The language exists to prevent the question *what does the player actually do?* from getting asked.

The test: Replace every noun and adjective in the pitch with a concrete example. If you can’t, the pitch has no content. If you can but the examples are boring, the concept is boring. The terminology was hiding that.

15.21. **Pillars that are wishes**

“Our three pillars are: deep combat, rich storytelling, and player freedom.” These are not pillars. They are Christmas lists. A pillar is a structural commitment that constrains what you build and what you cut. It should make some features impossible and other features mandatory. If a pillar is compatible with every possible design decision, it is not load-bearing. It is decoration.

The test: For each pillar, name one feature the team *wanted* to build that the pillar prohibits. If you can’t, the pillar permits everything, which means it supports nothing.

15.22. **Appeals to authority**

“The Witcher did it.” “Breath of the Wild proved this works.” “Miyamoto said…” A successful reference is not an argument. It is a way to stop the argument. The referenced game succeeded for specific structural reasons in a specific market context with a specific team. Citing it tells you nothing about whether *your* concept has the same structural properties. It tells you the pitcher has taste. Taste is not a plan.

The test: Remove every reference to another game or designer. Does the argument still stand? If it collapses, there was no argument. There was a borrowed reputation.

15.23. **“Just” anything**

“We just need to nail the combat feel.” “It’s just a matter of getting the tutorialization right.” “Just” is a magic word that makes unsolved problems disappear from the pitch. Every “just” hides a load-bearing assumption that has not been tested. “Just nail the feel” means the entire game depends on a craft outcome nobody in the room can guarantee or schedule.

The test: Replace “just” with “and we don’t yet know how to.” If the sentence now sounds alarming, the “just” was hiding something alarming.

15.24. **A-but-shaped-like-B**

“It’s a city builder but the city is a body.” “It’s a roguelike but every run is a therapy session.” The formula: take a proven structure (A) and dress it in an unexpected skin (B). The pitch sounds creative because A and B are surprising together. But surprise is not design. The question is whether A’s structure and B’s meaning produce something that couldn’t exist without both. Usually they don’t. Usually B is a theme and A is doing all the work, and the theme will be stripped out by month six because it creates friction with the loop, not reinforcement.

The test: Describe the game without mentioning B. Is it still interesting? If yes, B is a costume. If no, explain specifically how B changes the decisions the player makes inside A’s loop. “It changes the mood” is not an answer. “It changes which options are available and what the tradeoffs mean” might be.

15.25. **Wishful thinking**

The concept document explains why the game *can* work. It never explains why it *must* work given the specific structural properties of the subject. “Players will find the trading system compelling because trade is inherently interesting.” Will they? Trading is interesting to traders. Trading is a spreadsheet to everyone else. The argument needs to show that the *specific trading loop in this game* produces decisions that are interesting on their own terms, without the player already caring about trade.

The test: Assume the player has zero interest in your subject. Does the loop still produce interesting decisions? If the argument depends on the player arriving pre-motivated, the concept is borrowing interest it hasn't earned.

15.26. **Retrofitted arguments**

The concept was chosen before the analysis. The pitch was sold before the gates were applied. Now the gates are being applied backward — not to test the concept, but to justify it. You can tell because every gate "passes" on the first try, every concern has a pre-written answer, and the document reads like a defense brief, not an investigation.

The test: Look for surprise. A genuine gate analysis should produce at least one moment where the designer says "I didn't expect that" or "this is harder than I thought." If the document contains no surprises, it was written to confirm a conclusion, not to reach one. The strongest tell: the rejection criteria section is absent or perfunctory.

15.27. **Pandering**

The concept is shaped by what the audience (or the stakeholder, or the platform holder) is believed to want, rather than by what the subject structurally produces. This can be blatant ("battle royale is trending, so we add a battle royale mode") or subtle ("we noticed our audience responds to base-building, so we're integrating base-building into the loop"). Subtle pandering is harder to catch because it sounds like market awareness. The difference: market awareness informs which *subjects* to explore. Pandering deforms the *structure* of a concept to chase approval.

The test: Remove the pandering feature. Does the concept get stronger, weaker, or stay the same? If stronger or the same, the feature was appeasement. If weaker, check whether the weakness was already there and the feature was masking it.

15.28. **Generated concepts without judgment**

An LLM can produce a concept document that passes a surface reading of every gate. The structure will be present. The examples

will be plausible. And the concept will be dead on arrival, because no human decided this was a game worth making based on something they understood about the subject that the model doesn't.

The tell is uniformity. Generated concepts have no rough edges, no stubborn insistence on a specific detail that doesn't quite fit, no sign that someone fought for a decision against easier alternatives. They are smooth because nothing resisted during their creation. Real concepts have grain.

The test: Ask the designer to name the one thing about this concept they are *not* willing to negotiate on and *why*. A human with conviction will answer instantly and the answer will be specific. A laundered concept produces a pause, then a restatement of the pitch.

15.29. **No failure discussion**

The concept document contains no section on how the game might fail. No counterfactuals. No "if X doesn't work, the whole thing collapses." This is not confidence. It is avoidance. Every concept has failure modes. A designer who has thought seriously about a concept knows where the weak joints are and can name them. A designer who hasn't — or who has been told not to — produces a document that reads like a guarantee.

The test: Ask "what kills this game?" If the designer can't answer in one sentence, they haven't thought about it. If they answer with an external factor ("if the market shifts," "if we don't get the IP"), they're deflecting. The answer should be structural: "if the primitive turns out to be shallow after twenty repetitions, nothing else in the design saves it."

15.30. **Fragile concepts**

The concept requires every load-bearing component to work at full strength. The loop depends on the AI being good enough. The escalation depends on the content pipeline delivering on schedule. The intent depends on a narrative wrapper that hasn't been written yet. If any single element underperforms, the game fails, because no other element compensates.

Strong concepts have redundancy. The loop is interesting even if the AI is mediocre. The primitive still works if the content is thinner than planned. The intent comes through even without the narrative framing. A concept that requires perfection across the board is not ambitious. It is fragile.

The test: Degrade each load-bearing component by 30%. Does the game still work? If removing a third of the AI quality, or a third of the content, or a third of the visual fidelity collapses the experience, the concept has no structural margin. It is betting everything on execution, and execution is the one thing a concept document cannot guarantee.

15.31. **Misidentified audience**

The concept assumes a player who doesn't exist in sufficient numbers, or who exists but doesn't want what the game offers. "Hardcore strategy players who also want deep narrative and relaxing pacing." That player is real. There are forty of them. The concept is optimized for an intersection of tastes so narrow that even strong execution can't find a viable market. This is distinct from niche design, which targets a small audience on purpose. Misidentified audience targets a large audience and describes a small one by accident.

The test: Describe the target player without using genre labels. Describe what they do on a Tuesday night, what games they played last month, what they'd pick this game up instead of. If the description sounds like a specific person, you might have an audience. If it sounds like a demographic composite, you have a slide for investors.

15.32. **The untested hook**

The concept depends on a single moment — a reveal, a twist, a first-contact sensation — that cannot be repeated. The pitch deck shows the moment. The prototype demonstrates the moment. Everyone in the room is excited about the moment. But the moment is not the game. The game is what happens in the other forty hours. If the hook is the best part and the loop is the rest, the game will review well in the first paragraph and poorly in the last.

The test: Describe the game starting from hour three. If the description is less compelling than the pitch, the pitch is selling the hook, not the game.

---

16. **Related essays**

- *Making the Thing (https://gerolds.github.io/posts/making-the-thing/)*: Once you have a viable concept, how to build around a governing commitment.
- *Finding the Commitment (https://gerolds.github.io/posts/finding-the-commitment/)*: Why sticking with the commitment matters more than finding the “right” one.
- *Prototyping the Loop (https://gerolds.github.io/posts/prototyping-the-loop/)*: Searching for the invariant that makes the primitive feel alive.
- *The Elements of a Good and Widely Appealing Game (https://gerolds.github.io/posts/wide-appeal/)*: Ten axes that predict whether a finished game will travel.
- *The Studio Primitive (https://gerolds.github.io/posts/studio-os/)*: The operational loop (Promise → Proof → Cut → Ship) that frames when and how proof happens.
- *Studio DNA (https://gerolds.github.io/posts/studio-dna/)*: Why different teams are adapted for different loop classes and primitives.

Drafting assistance: Claude Opus. All claims mine; errors my responsibility.