WHITE PAPER

Aurora / Platinum White Paper

A hosted, human-readable home for the living white paper: what this project is trying to prove, how Platinum and the games fit together, why the conformance program exists, and how the narrative is versioned over time.

Current goal: Turn the current draft into a durable v1 release asset for a broad technical and builder audience: keep the main paper readable, connect it directly to shipped evidence and release discipline, maintain a printable PDF with explicit version/date metadata, and keep related-work and reviewer-minded checks part of the same release surface.

WHITE PAPER VERSION	UPDATED	CURRENT RELEASE	LANE	WHITE PAPER STATUS
v0.4.1-draft	2026-06-07	1.4.1.1	development	living white paper

BUILT	LATEST NOTE
June 12, 2026	1.4.1 production quality release

White Paper Project Area

Where this work lives in the repo and how it should be maintained.

Living Draft

The current narrative source of truth is `WHITE_PAPER.md`. It should keep moving with the project instead of becoming a one-off summary.

Project Area

The `white-paper/` directory is the durable project area for supporting material: the working-area README, citation ledger, and versioned white-paper releases.

Release Memory

Meaningful white-paper revisions should be snapshotted under `white-paper/releases/` with the Markdown, PDF, and PDF metadata preserved together so the narrative can be compared over time just like the software.

Hosted Surface

This generated page is the ongoing hosted, human-readable form. It should stay wired into the same documentation family as the project guide, platform guide, player guide, and release dashboard.

Reader Shape

Who this page is for and how it should balance readability with depth.

Broad Technical Reader

Assume curiosity, technical interest, and some builder intuition, but do not assume deep platform, browser, arcade, or conformance-program expertise.

Readable Main Narrative

The white paper should explain the thesis, architecture, release discipline, evidence program, and AI method clearly without becoming a complete internal manual.

Deeper Surfaces Nearby

Hosted guides, dashboards, release notes, and source docs should provide the extra detail so the main paper can stay focused and persuasive.

Hosted Entry Points

The main ways a reader should enter the white-paper story and related project surfaces.

Current White Paper Draft

Jump directly to the current rendered white paper inside this hosted page.

Citation Ledger

See the maintained list of outside ideas, reference families, and methodological influences.

Current White Paper PDF

Open the current lane PDF version of the white paper with explicit version/date metadata.

Project Overview Deck

Open the generated talk-through slide overview that travels beside the white paper in each release lane.

Recurring Project Roles

Read the updateable role definitions for manager, developer, architect, release authority, review, conformance, security, audio, and documentation automation.

Overview Slide Metadata

Inspect the generated metadata for the current lane slide overview artifact.

PDF Metadata

Inspect the generated metadata for the current lane PDF artifact.

Project Guide

Return to the broader hosted project map for Platinum, Aurora, Galaxy Guardians, ingestion, conformance, and release state.

Current Workstream Alignment

Open the current cross-thread priority map for Aurora challenge stages, Guardians v1, personas, ingestion, platform boundaries, and release docs.

Release Schedule And Issue Spine

Open the forward-looking release-family schedule that maps workstreams, GitHub issues, documentation, and lane gates.

Lane Project Page

Read the shorter public-facing project summary for the current lane.

Conformance Dashboard

Open the live score, confidence, cost, and investment dashboard when the narrative needs the measured readout.

White Paper Project Source

Open the repo-owned white-paper project area in GitHub.

White Paper Markdown Source

Open the current Markdown source directly.

Review Posture

How the white paper should be reviewed as a release surface instead of treated like an unreviewed note.

HTML Matters

The hosted page should read well on desktop and mobile, keep diagrams and screenshots legible, and avoid accidental raw Markdown leakage or repetitive section structure.

PDF Matters

The printable/exportable PDF should carry explicit version/date information and avoid bad page breaks, oversized dark backgrounds, or weak diagram reproduction.

Reviewer Mentality

Reviewers should tighten repetition, question weak claims, verify references, and treat the white paper as part of the release promise rather than as detached commentary.

Evidence Drift Check

The white-paper review spine should also verify preserved-source integrity and catch stale source-path drift in active evidence docs, not only PDF formatting issues.

Recurring Project Roles

The human, agent, machine, and build-process roles that make the operating model assignable and reviewable.

Roles Are Explicit

Manager, developer, architect, release authority, parallel worker, security, code review, conformance, audio, player-visible review, and documentation generation roles are now named instead of implied.

Definitions Are Maintainable

`white-paper/PROJECT_ROLES.md` is the updateable role source. It records definition, invocation timing, automation status, and links to detailed repo sources.

Automation Has Boundaries

Build and harness automation can refresh evidence and block weak releases, but beta/production authority and player-visible quality judgment remain explicit human decisions.

Role Definitions

Open the rendered role table and maintenance notes.

Current Project State

See the active project state used to orient new sessions.

Multi-Machine Allocation

See how MacBook release authority and iMac parallel work are separated.

White Paper Release History

The first durable release pattern for preserving the narrative over time.

v0.1.0 Snapshot

The first seeded snapshot of the white paper in browsable release form.

v0.1.0 Release Notes

Short note describing what the first white-paper release established and what should happen next.

V1 Release Path

The next best steps for turning the current seed into a true v1 white-paper release.

- Tighten the audience voice so the v1 paper reads clearly as a public-facing promotional and methodological document, not only as an internal orientation memo.
- Recover and cite the earlier Karpathy-style assessment directly so the methodology section can point to a precise prior observation instead of only a conceptual placeholder.
- Add selected screenshots or rendered excerpts from the hosted dashboard, project page, and game surfaces so the paper shows the artifacts it talks about.
- Add a compact internal bibliography that points readers to the canonical source docs: project state, ingestion framework, conformance economics, release policy, and application/platform guides.
- Sharpen the historical arc from 1.0.0 launch to 1.2.0 platform framing to 1.4.0 multi-game and conformance maturity so the story reads as a clear progression.
- Decide which claims should be explicitly public and promotional versus which should remain engineering-facing detail, then trim or regroup sections accordingly.
- Connect the v1 paper to one stable release-quality evidence pack so the narrative and the measurable artifacts can be reviewed side by side.
- Keep the hosted HTML page and lane PDF under the same reviewer checklist so presentation quality, page breaks, and visual clarity improve intentionally over time.
- Refresh the related-work log periodically with focused external searches and short relevance notes instead of letting public references accumulate haphazardly.
- Add a concise section on what this project teaches about generative-AI-assisted software practice beyond games, with examples tied to actual repo artifacts.
- Cut the first intentional white-paper v1 snapshot under `white-paper/releases/` once the citations, visuals, and public voice are all coherent enough to stand alone.

Current White Paper Draft

The living white paper as it exists in the repo today, rendered into the same hosted-document family as the other project guides.

Generated from `WHITE_PAPER.md` during build.

Status: living white paper Current draft: `v0.4.1-draft` Date: `2026-06-07` Audience: broad technical readers, interested builders, collaborators, future reviewers, and public-facing project storytelling

This document is the maintained narrative explanation of what this project is doing, why it is being built this way, and how the approach evolves over time. It is intended to be both a promotional piece and a disciplined reminder to the team: the software, the evidence program, the release process, and the generative-AI workflow all matter together.

See also:

- white-paper/README.md
- white-paper/CITATION_LEDGER.md
- white-paper/REVIEW_CADENCE.md

Private companion store only; not exposed from the public repo.

Overview / Preamble

This project is not only a browser game repo.

It is a deliberate attempt to build a professional software program around a harder claim:

- that generative AI can help produce serious, well-released software
- that aggressive iteration does not need to mean careless iteration
- that reference-driven quality can be pursued with explicit evidence instead

of vague taste alone

- that local harnesses, release lanes, review packets, and durable artifacts can

keep AI-assisted work honest

`Platinum` is the reusable browser-arcade host. `Aurora Galactica` is the first shipped application on that host. `Galaxy Guardians` is the second-game proof that the platform, ingestion program, and conformance discipline can grow beyond a single title.

The larger point is not "we used AI to make a game."

The larger point is that we are building a system in which:

- the product is real
- the releases are real
- the documentation is real
- the tests and harnesses are real
- the evidence is real
- and the model-assisted work is useful because it is forced to leave behind

rerunnable artifacts

Section Overview

- `1. Thesis`: what this project is trying to prove.
- `2. Program snapshot`: where Platinum, Aurora, and Galaxy Guardians stand

right now.

- `3. Five-layer operating model`: platform, games, ingestion, harnesses, and

release economics as one program.

- `4. Ingestion strategy`: how external evidence becomes structured game truth.
- `5. Challenge-stage ingestion case study`: how richer reference recovery changed the plan for Aurora's hardest gameplay gap.
- `6. Harnessing and conformance`: how we measure quality instead of asserting it.
- `7. Release discipline`: how dev, beta, and production remain explicit and professional.
- `8. Generative AI role`: how model work accelerates the project without replacing evidence.
- `9. Working loop`: how the project turns a gap into evidence, implementation, measurement, and release learning.
- `10. Historical evolution`: how the project moved from launch to platform to multi-game conformance.
- `11. Citation program`: how outside ideas and source recovery work should be tracked explicitly.
- `12. Related work`: how outside agent/evaluator work informs the project.
- `13. Internal canonical docs`: how this paper stays short without losing traceability.
- `14. Why this project matters`: why the project is larger than a game repo.
- `15. Living-paper policy`: how this white paper should be maintained and released over time.

How To Read This Paper

This page is meant to be the readable narrative layer, not the whole archive.

- Read this paper for the story, the method, and the architectural shape of the project.
- Use the diagrams, screenshots, and charts here as anchors, not as the full evidence pack.
- If you want implementation detail, detailed metrics, or release-by-release operational context, jump into the linked hosted documentation rather than making this paper carry everything.

Useful deeper surfaces:

- project-guide.html
- project-guide.html#release-schedule-spine

- platinum-guide.html
- application-guide.html
- conformance-dashboard.html
- release-dashboard.html
- release-notes.html
- white-paper.pdf

First Draft

1. Thesis

The core thesis of this project is that generative-AI-assisted software can be built aggressively without becoming hand-wavy, fragile, or unprofessional.

That requires a few non-negotiable rules:

- the software must ship as a real public product, not just as a demo
- improvements must be tied to evidence whenever the question is measurable
- platform boundaries must stay explicit so reuse is deliberate rather than

accidental

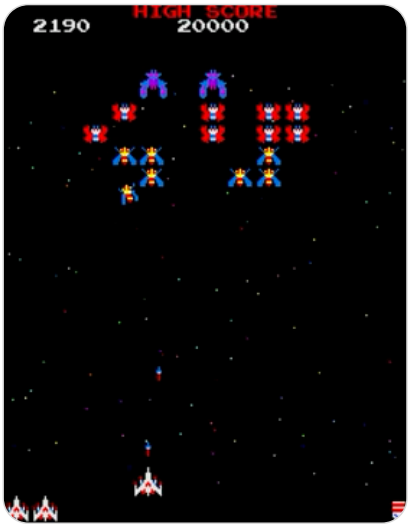
- release claims must be backed by committed docs, review artifacts, and

rerunnable checks

- model work should increase leverage, but repeated assessment should

increasingly move into local CPU/browser harnesses

This is why Platinum, Aurora, Galaxy Guardians, ingestion, harnesses, scorecards, review packets, and release notes belong in the same story.



The image above is intentionally simple: it reminds the reader that all of the process, evidence, and release discipline in this paper exist in service of a real playable artifact, not only a methodology exercise.

Further detail:

- application-guide.html

- release-notes.html

2. Program Snapshot

As of `2026-06-07`, the project can be described in one page:

AREA	CURRENT ROLE	WHY IT MATTERS
`Platinum`	Shipped browser-arcade host platform	Proves that reusable shell, services, lane model, and release discipline can exist without absorbing game-specific truth.
`Aurora Galactica`	First shipped playable Platinum application	Serves as the strongest current proof that the platform can host a real public game and improve its conformance over time.
`Galaxy Guardians`	Preview-first second-game and first-class ingestion/conformance target	Proves that the platform and the evidence program can support a second game without simply cloning Aurora.
Ingestion framework	Source-to-structured-evidence pipeline	Keeps new-game and fidelity work anchored in manifests, clips, event logs, waveforms, contact sheets, and provenance.
Harness and conformance system	Scorecards, correspondence checks, dashboards, and gates	Turns quality claims into measurable, reviewable outputs.
Release and economics program	Lane discipline, review packets, docs refresh, local-vs-cloud resource accounting	Makes the project look and behave like a professional release program rather than an endless prototype.

Current maintained metric read:

SCOPE	CURRENT READ	INTERPRETATION
Project conformance economics	`8.7/10` roll-up	Strong broad score, but the next release value depends on closing the worst rows rather than polishing the average.
Application artifact conformance	`7.46/10`	The weakest row is `impact-explosion-visual-feedback`, so damage, hit, loss, and explosion feedback remain a major user-experience target.
`Aurora` challenge-stage set pieces	`4.3/10` strict score	The clearest gameplay-conformance blocker: movement, graphics, alien novelty, and target-video fit are still far from mature Galaga-like bonus exhibitions.
`Aurora` challenge grammar readiness	`25/25` reference-backed first-five group contracts; `8.6/10` control readiness	Analysis is now ahead of runtime implementation; the next useful work is promotion-safe movement grammar, not more broad planning.
`Galaxy Guardians` long-surface/persona review	`7.0/10`	Credible second-game process proof, but not yet a production-mature public game; v1 needs opening-slice quality, score/result identity, platform parity, and Watch/Rival/persona reuse.
Resource/economics ledger	`904` measured runs, `58,277s` tracked wall time, `58,392s` CPU time, about `1.48GB` artifact accounting	Shows the operating doctrine: turn model-assisted insight into local CPU/browser harnesses and track the cost of quality movement.

The evidence program also became more concrete in the latest pass:

- release authority moved from `iMacM1` to this MacBook M4 for the current

release path

- release conformance artifacts, the game conformance catalog, and white-paper

PDF metadata are refreshed enough for `publish:check:dev`

- hosted `/dev` now carries the current `1.4.0.1` forward-review line,

including the consolidated Aurora challenge grammar, Guardians ingestion/conformance cleanup, refreshed dashboards, public project guide, white-paper PDF, slides, release-schedule spine, and review packet

- the current cross-thread priority map is preserved in

PROJECT_WIDE_WORKSTREAM_ALIGNMENT_2026-06-07.md

- the June 1 preserved-source expansion added richer Galaga, Galaxian, and

Space Invaders evidence lanes, including manuals, strategy/walkthrough bundles, sprite/cue packages, challenge-stage videos, and cabinet/spec references

- the public/private artifact boundary is now explicit: source metadata and

summaries stay in this repo, while copied or derived source bytes belong in the companion private artifact store

Private companion store only; not exposed from the public repo.

Private companion store only; not exposed from the public repo.

These pack views help a broad reader understand one of the project's central claims: `Aurora Galactica` and `Galaxy Guardians` are not supposed to be two skins on one game. They are meant to be separate applications living on one host platform.

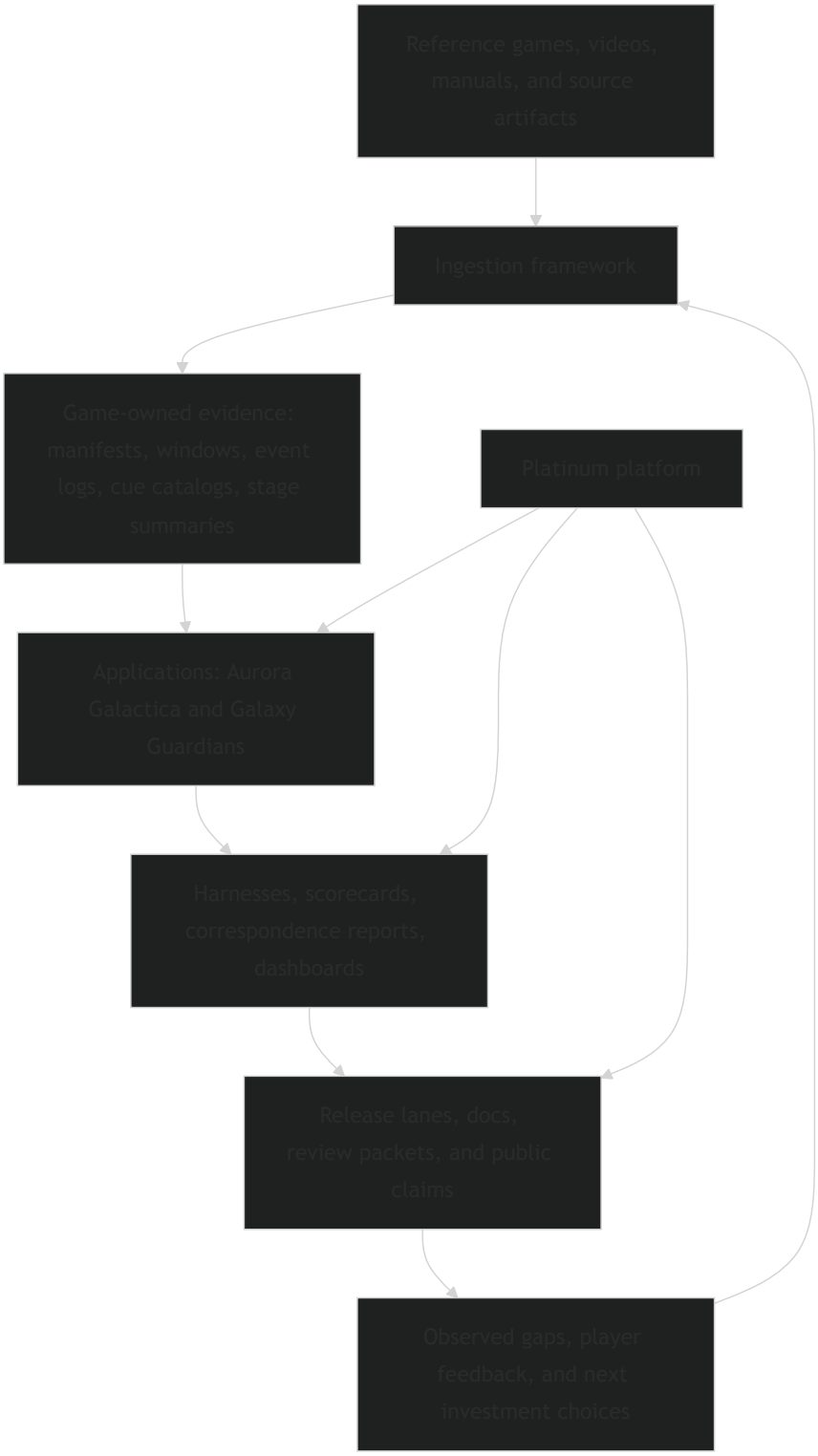
> TODO illustration: Choose a small three-panel progression strip that shows how the public face of the project evolved from `1.0.0` launch to `1.2.0` Platinum framing to `1.4.0` multi-game posture. The most illustrative version may be gameplay first, shell first, or docs/release-surface first, and we should pick that deliberately rather than guessing.

Further detail:

- project-guide.html
- platinum-guide.html

3. Five-Layer Operating Model

The repo already describes the work as a layered system. The white paper should make that model legible at a glance.



Private companion store only; not exposed from the public repo.

Private companion store only; not exposed from the public repo.

The important discipline is separation of ownership:

- `Platinum` owns shell, hosting, shared services, contracts, and release framing.

- Games own rules, scoring, progression, audiovisual identity, and their own

conformance truth.

- Ingestion owns provenance and structured evidence.
- Harnesses own repeatable evaluation.
- Release discipline owns the public promise.

When those layers blur, the project becomes harder to explain, harder to test, and easier to accidentally fake.

Further detail:

- project-guide.html#platform-vs-applications
- platinum-guide.html
- PROJECT_STATE_AND_CONFORMANCE_PROGRAM.md

4. Ingestion Strategy

Ingestion is the front half of engineering, not a side notebook.

The project does not want new games or fidelity improvements to come mainly from memory, vibes, or post-hoc rationalization. Instead, it wants evidence to arrive in structured forms that can be reused:

- source manifests
- preserved clips and windows
- frame contact sheets
- waveforms and spectrograms
- reference-side event logs
- semantic annotations
- confidence notes
- scorer and correspondence targets

For `Aurora`, this keeps Galaga-like timing, audio, pressure, and stage-shape questions grounded in real artifacts.

For `Galaxy Guardians`, ingestion matters even more. It is the mechanism that prevents the second game from turning into "Aurora with different labels." The game should become more complete by promoting `Galaxian` evidence into game-owned scoring, wave timing, sprite identity, audio expectations, and runtime correspondence checks.

In short:

- external artifacts teach the game
- ingestion turns artifacts into structured evidence
- the application implements against that evidence
- Platinum stays the host rather than the hidden author of the game

Private companion store only; not exposed from the public repo.

Private companion store only; not exposed from the public repo.

These reference contact sheets are useful because they show the project's ingestion claim in a form a non-expert can understand quickly. We are not only describing classic arcade behavior; we are collecting windows, studying them, and turning them into reusable evidence.

That claim is now easier to defend concretely because the repo carries preserved-source lanes as well as derived analyses. The current reference inventory includes Galaga audio cue packs, Galaga challenge-stage videos, StrategyWiki sprite/walkthrough bundles, arcade-museum cabinet/spec pages, Galaxian no-voiceover and full-session gameplay, Galaxian FLAC cue packs, Galaxian operator/manual material, and early Space Invaders intake packages. The project is moving source recovery out of memory and into committed provenance, with copied media bytes separated into the companion private artifact store when public-hosting would be inappropriate.

The important process upgrade is that ingestion now has required outputs, not only nice-to-have research notes. A serious game line should maintain:

- an alien/enemy index
- an audio cue index
- a stage or wave index
- a persona index
- sprite/runtime scale evidence
- entry choreography evidence
- target-artifact coverage before major implementation claims

That makes the second and third games less likely to inherit Aurora-specific assumptions by accident.

> TODO illustration: Pick the single best “ingestion in action” image for v1. The strongest option might be a contact sheet, a waveform-plus-contact-sheet pair, or a staged comparison between raw source footage and the structured artifact family that comes out of it.

Further detail:

- project-guide.html#classic-arcade-ingestion-framework-doc
- application-guide.html
- CLASSIC_ARCADE_INGESTION_FRAMEWORK.md
- reference-artifacts/preserved-sources/README.md

5. Challenge-Stage Ingestion Case Study

Aurora's challenge stages are the clearest example of why the project had to get more serious about ingestion and annotation.

The user-visible complaint was simple: the challenge stages did not feel like classic Galaga-style bonus exhibitions. They were safe, and some broad coverage checks passed, but they lacked the thing that players actually learn and remember: coherent group arrivals, varied alien families, readable scoreable lanes, stage-to-stage novelty, and the sense that each challenge is a designed set piece rather than a generic wave.

That exposed a weakness in the earlier measurement model. Old diagnostics could say that challenge coverage existed because enemies appeared, did not shoot, and followed some path families. That was too generous. The stricter model now starts the player-facing challenge-stage read from a harsh baseline and asks a more useful question: does this stage create the same kind of spectacle, movement memory, and perfect-score opportunity as the target examples?

Recent ingestion and annotation work changed the situation in four ways:

UPGRADE	WHAT CHANGED	IMPACT SO FAR
Reference recovery	User-supplied and preserved Galaga challenge compilations now provide media-backed windows for the tracked challenge family.	The bottleneck moved from "find examples" to "label and implement against examples."
Stage labeling	The project now treats stages as ordinary play stages and names bonus windows as `Challenging Stage 3-4`, `Challenging Stage 7-8`, and so on.	Human review, docs, harnesses, and developer tools have clearer shared language.
Object-track analysis	CPU object tracking converts challenge clips into per-group target vectors: entry side, timing, path range, lower-field travel, and path-family hints.	Target-track readiness and control readiness are now around `8.6/10`, giving implementation a concrete target shape.
Candidate guards	Runtime candidates are checked against target-video fit and human-perfect potential before promotion.	The process has already prevented bad promotions, including a Stage 3 candidate that slightly improved expected-label fit but reduced human-perfect potential by `1.6/10`.

The honest current read is mixed.

On the positive side, the challenge-stage work has improved the project faster than a purely subjective tuning pass could have. The latest target structure covers `8` tracked challenge windows and `40` reference-backed groups. The harness can generate paired target-vs-current videos from stage start, contact sheets, timing drift summaries, target trajectory controls, and candidate before/after reports. That is a large process gain, and it should make future Aurora, Galaxy Guardians, and third-game work cheaper and less guessy.

On the negative side, the same evidence makes the gameplay gap harder to hide. The strict challenge-stage score is still only about `4.3/10`: movement `4.2/10`, graphics `4.5/10`, alien novelty `3.9/10`, target-video object-track fit `3.6/10`, and zero release-ready challenge contracts. The no-shot and no-ship-loss safety rule is strong, but safety is now treated as a guardrail, not as proof of conformance. A safe challenge stage can still be boring, visually weak, or badly paced.

That distinction matters for the project's AI-assisted method. This work is a success as ingestion, annotation, and evaluator-building. It is not yet a success as shipped player experience. The next phase must convert the evidence into runtime movement grammar that can produce better stages without endless manual special cases.

The next-work categories are therefore specific:

1. Promote five-group labels for each challenge window: first visible frame,

entry side, exit side, path family, scoreable band, alien family, and perfect-bonus opportunity.

1. Build a reusable movement-grammar layer that represents group arrivals,

arcs, loops, ladders, hooks, crossings, serpentine paths, and exits as editable contracts rather than one-off constants.

1. Keep a human-perfect guard in the candidate loop so conformance does not

improve by making stages less playable or less learnable.

1. Add active sprite-motion evidence for challenge aliens, because static

sprite crops do not capture flapping, pulsing, dive poses, or specialty target identity.

1. Use paired target/current clips and contact sheets as review artifacts

whenever a challenge-stage change is claimed.

1. Generalize the same grammar to normal-stage entry behavior, not only

challenge stages, so the platform can support game-specific variation without hard-coding Aurora's current patterns into Platinum.

The reason this should speed quality improvement is that it changes the shape of the work. Instead of asking the model or a human to "make the stage feel more Galaga-like," the system can ask a narrower question: which group contract is missing, which trajectory differs, which alien family is wrong, and which candidate improves target fit without reducing perfect-score readability?

Further detail:

- application-guide.html#challenge-stage-conformance
- GALAGA_TARGET_ARTIFACT_COVERAGE.md
- LEVEL_VISUAL_TIMING_ALIGNMENT.md
- AURORA_SPRITE_MOTION_CORRESPONDENCE.md

6. Harnessing And Conformance

This project is serious about the difference between "better" and "better by a rerunnable measure."

The conformance system exists so that quality can be described with more precision than a mood:

- timing correspondence
- sequence correspondence
- outcome correspondence
- spatial correspondence
- visual correspondence
- audio correspondence
- persona and progression correspondence

The current Aurora scorecard turns this into a twelve-category quality model. That matters for two reasons.

First, it helps choose investments that are actually player-visible.

Second, it protects the team from false confidence. A `10/10` is explicitly not "perfect"; it means "maxed at current scorer resolution." Better evidence or a better evaluator can lower a score while making the project more truthful.

The harness program also stays intentionally classified:

- `platform` harnesses protect shell, hosting, docs, and shared services
- `application` harnesses protect game-specific rules and behavior
- `boundary` harnesses protect the seam between Platinum and the games

Representative committed commands in this strategy include:

- `npm run harness:measure`
- `npm run review:code`
- `npm run review:ledger`
- `npm run harness:check:galaxy-guardians-first-class-conformance`

This is the deeper quality claim of the project: bugs, polish, and release readiness should increasingly move from memory and opinion into explicit checks, artifacts, and dashboards.

Private companion store only; not exposed from the public repo.

Private companion store only; not exposed from the public repo.

The value of these charts is not only that they look rigorous. They show that the project tries to externalize quality questions into surfaces that can be inspected, debated, and rerun.

The newest dashboard makes the current prioritization uncomfortable in the right way. Basic challenge timing, combat response, capture/rescue rules, and several shell surfaces pass as guardrails. But the strict challenge-stage set-piece scorer is only `4.3/10`, with movement `4.2/10`, graphics `4.5/10`, novelty `3.9/10`, target-video object-track fit `3.6/10`, and zero release-ready challenge contracts. That score is not a failure of the process. It is the process doing its job: replacing a too-generous broad proxy with a more honest stage-by-stage conformance read.

Further detail:

- conformance-dashboard.html
- project-guide.html#release-conformance-dashboard-doc
- project-guide.html#conformance-economics-doc

7. Release Discipline And Professionalism

The project treats release engineering as part of product quality.

That means the release lanes are not cosmetic:

1. local `localhost`
2. hosted `/dev`
3. hosted `/beta`
4. hosted `/production`

Each lane carries a different stability promise, documentation expectation, and testing posture. The project is intentionally trying to behave like a software program with real public accountability:

- release notes are first-class
- docs are part of the release surface
- review packets and review-learning ledgers are durable evidence
- the white paper must read well both as hosted HTML and as printable PDF
- production claims should come from an approved beta lineage
- major releases should refresh dashboards, scorecards, and strategic docs

This matters because AI-assisted speed is only impressive if the public result still feels trustworthy.

The "reviewer" mentality should therefore be explicit. The paper is not done just because the words are present. The release surface should also be reviewed for lane coherence, build metadata, conformance freshness, and historical path drift.

As of this draft, the current production recommendation is deliberately conservative:

- hosted `/production` remains the stable public `1.4.0` line
- hosted `/dev` and `/beta` are review lanes for the next candidate family
- the MacBook M4 is now release authority
- the current candidate is suitable for dev-lane review after preflight, but

not yet a new `1.4.1` production promise

- challenge-stage quality and documentation consistency remain the two clearest

reasons to defer production

That restraint is part of the method. The project should not treat a passing publish script as the same thing as a strong public release story.

The reviewer pass should keep looking for:

- repeated ideas that can be tightened
- diagrams or images that create awkward whitespace or weak page breaks
- TODOs that are unclear or more revealing of indecision than of real planning
- HTML reading flow on desktop and mobile
- PDF export quality, especially around image scale, table breaks, and diagram

legibility

Further detail:

- release-dashboard.html
- release-notes.html
- project-guide.html#testing-doc
- project-guide.html#code-review-model-doc
- white-paper/REVIEWER_CHECKLIST.md
- white-paper/REVIEW_CADENCE.md

8. How Generative AI Fits

The project does use generative AI heavily, but not as a substitute for engineering structure.

The intended operating doctrine is:

- use models to design evaluators, synthesize options, write code, review code,

summarize evidence, and tighten the next decision

- use local CPU/browser harnesses for repeated measurement, sweeps, and

regression checks

- convert model-assisted insight into committed repo artifacts whenever

possible

- log resource use and compare spend against score movement
- keep human review, public release notes, and versioned documentation in the

loop

The repo already describes one part of this explicitly as a "Karpathy-loop-like" pattern:

- inspect concrete examples
- improve the dataset and evaluator
- make a small candidate change
- run it

- study failures
- fold the learning back into the system

That is a strong fit for the broader project identity. The point is not merely to ask a model for code. The point is to build a system in which model help leaves behind better evaluators, better artifacts, and cheaper future decisions.

Private companion store only; not exposed from the public repo.

Private companion store only; not exposed from the public repo.

These charts help keep the AI story grounded. The point is not only that model assistance exists; it is that the project is trying to compare that assistance with local repeatable measurement and with visible quality movement.

The current economics ledger is intentionally imperfect but already useful:

- `904` measured runs are logged
- local CPU accounts for `576.5` tracked wall minutes
- browser-backed local work accounts for `429.6` tracked wall minutes
- declared GPU-equivalent/Codex/model/API work accounts for `630.8` tracked

wall minutes, but remains under-instrumented and partly overlapping by design

- audio work consumed the largest local-compute block, while gameplay complexity

and stage arc account for the largest positive score movement

This is exactly the planning tension the project wants to expose. If audio keeps consuming large compute blocks for modest score movement, the next investment should either improve the audio evaluator itself or shift energy to the higher-value challenge-stage movement grammar.

Further detail:

- project-guide.html#conformance-economics-doc
- CONFORMANCE_ECONOMICS.md

9. Working Loop

The operating loop of this project is more important than any single feature.

This loop explains how the project tries to be both aggressive and controlled. The aggressiveness comes from fast iteration and model-assisted leverage. The control comes from evidence, harnesses, explicit ownership boundaries, and release discipline.

> TODO illustration: Add one compact “question -> evidence -> harness -> change -> rerun” visual from a real case study. Audio cue alignment, stage-opening timing, or a Galaxy Guardians reference-promotion slice are the strongest current candidates, but we should choose the one that is most legible to a broad reader.

10. Historical Evolution So Far

The release notes already show a clear arc, and the white paper should make it easy to retell.

RELEASE	MEANING	STRATEGIC SHIFT
`1.0.0`	First public Aurora launch	The project became a real public product with live scoring, pilot identity, replay visibility, and a real release ladder.
`1.2.0`	Platinum Release 1	Aurora was reframed as the first application on a reusable platform, making platform/application separation explicit.
`1.4.0`	Current multi-game and conformance baseline	The public line now carries stronger documentation, review evidence, persona/replay follow-through, and a clearer Galaxy Guardians posture.

This means the project has already moved through three meaningful phases:

1. prove a public game can ship
2. prove the host platform is real
3. prove multi-game and conformance maturity can become part of the release

identity

The next phase should be to prove that this method scales:

- deeper Aurora conformance
- stronger Galaxy Guardians first-class completeness
- cleaner shared Platinum operations
- more reusable ingestion and assessment infrastructure
- a third-game intake path, currently represented by Space Invaders / Windigo

Invaders preserved-source and planning lanes

> TODO illustration: Build a release-history gallery with one screenshot or architectural surface per milestone. The current paper names the milestones clearly, but a short visual strip would make the progression easier to absorb at a glance.

Further detail:

- release-notes.html
- project-guide.html#release-note-140-beta-1-doc
- project-guide.html#release-note-130-production-refresh-doc

11. Citation Program

This white paper should not quietly absorb ideas or source recovery work without naming them.

We want a maintained citation program that records:

- what outside work or source material influenced us
- how we used it
- what we learned from it
- what changed in the repo because of it
- what still needs to be recovered, linked, or tightened

The living ledger for that work starts here:

- white-paper/CITATION_LEDGER.md

The source-recovery side of that program now has a matching repo-owned surface:

- reference-artifacts/preserved-sources/README.md
- REFERENCE_MEDIA_INVENTORY.md

That matters because provenance is not only a footnote here. It is a release quality concern. If a timing study, audio comparison, or historical claim depends on a file that only exists in somebody's old downloads folder, the project is less professional than it looks.

The first open citation debt is the prior standalone assessment of the Karpathy-style research/evaluator loop. The repo contains the conceptual thread already, but the older assessment should be recovered and linked directly in a future white-paper release rather than reconstructed from memory.

Further detail:

- white-paper/CITATION_LEDGER.md

12. Related Work

This project should periodically stop and look outward.

The right pattern is not to stuff the paper with literature. The right pattern is to do focused searches, add high-signal sources, explain their relevance in plain language, and keep the public references linked to a maintained log.

Current seeded related-work set:

- Anthropic, "Building effective agents" (2024-12-19): relevant because it argues for simple, composable agent patterns and evaluator-optimizer loops rather than ornamental workflow complexity.
- Anthropic, "Writing effective tools for agents - with agents" (2025-09-11): relevant because our harnesses, scripts, and release tools are all explicit contracts between model assistance and deterministic system behavior.
- Anthropic, "Demystifying evals for AI agents" (2026-01-09): relevant because it reinforces our investment in transcripts, graders, repeated trials, and measurable evaluator quality instead of anecdotal confidence.
- Anthropic, "Trustworthy agents in practice" (2026-04-09): relevant because it frames guardrails, reviews, and operator-visible controls as part of the product surface, not only as implementation details.
- METR, "Measuring AI Ability to Complete Long Tasks" (2025-03-19): relevant because it gives a sharper lens on what kinds of work current agents can sustain autonomously, which is directly relevant to our preference for small evaluator-visible loops and local reruns over vague claims of full autonomy.
- OpenAI, "PaperBench" (2025-04-02): relevant because it shows how large agentic tasks become more reviewable when decomposed into explicit rubrics and many individually gradable subtasks, which is close to how our own harnesses and conformance categories create reviewable surfaces.

Maintained deeper log:

- white-paper/RELATED_WORK.md

13. Internal Canonical Docs

This paper should stay readable because the repo already has deeper canonical surfaces nearby.

The shortest list of internal references that best supports the claims here is:

- project-guide.html
- platinum-guide.html
- application-guide.html
- conformance-dashboard.html

- release-dashboard.html
- CLASSIC_ARCADE_INGESTION_FRAMEWORK.md
- PROJECT_STATE_AND_CONFORMANCE_PROGRAM.md
- CONFORMANCE_ECONOMICS.md
- REFERENCE_MEDIA_INVENTORY.md
- GAME_CONFORMANCE_CATALOG.md
- RELEASE_CONFORMANCE_DASHBOARD.md
- reference-artifacts/preserved-sources/README.md

If the main paper starts to feel long, that is usually a sign that one of these surfaces should carry more of the detail instead.

14. Why This Project Matters

The project matters because it is trying to demonstrate a concrete alternative to two weak extremes.

It is not:

- slow, ceremonial process that kills iteration
- or fast AI-assisted output that cannot explain or defend itself

Instead, it aims for a middle path:

- ambitious shipping
- real public releases
- strong platform boundaries
- evidence-backed improvement
- local-first repeated measurement
- model-assisted leverage
- versioned documentation and release storytelling

If that works, the result is more than a good arcade project. It becomes a useful pattern for how generative AI can participate in professional software work without dissolving quality standards.

This is also why the paper should remain readable. A broad technical reader does not need every source artifact inline. They need a coherent narrative, selected visual proof, and obvious places to go next if they want more depth.

15. Living White Paper Policy

This document should evolve the same way the project evolves: intentionally, versioned, and with historical memory preserved.

Working policy:

- `WHITE_PAPER.md` is the current living draft
- the hosted HTML and printable PDF should be kept aligned as two views of the

same maintained release surface

- meaningful revisions should be snapshotted under `white-paper/releases/`
- each snapshot should preserve the exact white paper text for that release
- when a release PDF exists, the snapshot should preserve the Markdown, PDF,

and generated PDF metadata together

- the citation ledger should be updated when outside work materially changes the

project story

- the related-work log should be refreshed periodically with focused online

searches and brief relevance commentary

- the recurring reviewer rhythm in `white-paper/REVIEW_CADENCE.md` should be

treated as normal maintenance, not as a one-off cleanup exercise

- reviewer-checklist expectations should be treated as part of release quality,

not optional cleanup

- every meaningful software release does not need a new white paper release, but

every strategic narrative shift probably does

Good triggers for a new white paper release:

- a major public release family
- a major shift in the conformance program
- a stronger second-game milestone
- a meaningful change in the AI/harness/evidence operating model
- recovery of an important citation or conceptual influence

Immediate Next Additions

- Recover and link the earlier Karpathy-style assessment if it exists outside

this repo.

- Add one deliberate progression gallery for milestone history and one deliberate

“evidence in action” case-study image once we decide which examples explain the project most clearly.

- Add a compact public diagram that shows the new source-to-metric pipeline:

preserved source package -> extracted window -> semantic event/crop/path target -> runtime capture -> conformance score -> release gate.

- Add a deeper table that compares Aurora, Galaxy Guardians, and Windigo by

ingestion maturity, not only by current playability.

- Decide which evidence families should be summarized publicly and which should

remain in the private artifact store behind public-safe metadata.

- Keep the HTML and PDF release surfaces under reviewer scrutiny so that spacing,

diagrams, repeated ideas, and print behavior all improve with the narrative.

- Keep the audience tuned for a broad technical and builder readership: assume

interest, assume intelligence, but do not assume deep prior expertise.

Citation Ledger

Maintained list of outside ideas, source families, how they were used, and what still needs to be recovered or tightened.

Generated from `white-paper/CITATION_LEDGER.md` during build.

This ledger tracks source families, methodological influences, and outside ideas that materially shape the project story.

The goal is not only to cite things. The goal is to record:

- what influenced the work
- how that influence showed up in the repo
- what the team learned from it
- what citation or linkage debt still remains

Status Key

- `linked`: the reference is represented clearly enough in the repo today
- `partial`: the idea is present, but the exact earlier note, external source,

or final public citation still needs to be recovered or tightened

- `queued`: important enough to track now, but not yet integrated well enough

to claim as a polished citation

Current Entries

REFERENCE	KIND	HOW WE USE IT	WHAT WE LEARNED	CURRENT REPO ANCHOR	STATUS
Karpathy-style evaluator loop and earlier project assessment	conceptual / methodological	Shapes the idea that we should inspect concrete examples, improve evaluators, make small candidate changes, rerun, and study failures instead of tuning only by opinion.	Better evaluators can be as important as better runtime code. A stricter scorer can lower a score while making the project more truthful.	`PROJECT_STATE_AND_CONFORMANCE_PROGRAM.md`, `CONFORMANCE_ECONOMICS.md`, `RELEASE_NOTE_1.3.0.1_HOSTED_DEV_REVIEW.md`	`partial`
Anthropic, "Building effective agents" (2024-12-19)	external methodological reference	Reinforces the idea that agentic systems should prefer simple, composable loops and explicit evaluator structures instead of ornamental complexity.	Simpler loops become more legible when the evaluator, harness, and release artifacts are visible to reviewers.	`WHITE_PAPER.md`, `white-paper/RELATED_WORK.md`, `PROJECT_STATE_AND_CONFORMANCE_PROGRAM.md`	`linked`
Anthropic, "Writing effective tools for agents - with agents" (2025-09-11)	external methodological reference	Supports our view that tools are explicit contracts and that agent quality depends heavily on the quality, shape, and reviewability of those tools.	Better tools and better tool descriptions are a form of product quality, not only implementation detail.	`WHITE_PAPER.md`, `white-paper/RELATED_WORK.md`, `TESTING_AND_RELEASE_GATES.md`	`linked`
Anthropic, "Demystifying evals for AI agents" (2026-01-09)	external evaluation reference	Supports our investment in repeated trials, transcripts, graders, and explicit evaluation design for agent-assisted work.	Evals become more useful when they are cheap to rerun, narrow in scope, and part of the everyday engineering loop.	`WHITE_PAPER.md`, `white-paper/RELATED_WORK.md`, `CONFORMANCE_ECONOMICS.md`	`linked`
Anthropic, "Trustworthy agents in practice" (2026-04-09)	external governance / operations reference	Aligns with our insistence that guardrails, human review, release notes, and reviewer-visible controls are part of the	Trustworthiness is easier to discuss honestly when it is backed by durable checks and visible	`WHITE_PAPER.md`, `white-paper/RELATED_WORK.md`, `RELEASE_POLICY.md`, `CODE_REVIEW_MODEL.md`	`linked`

REFERENCE	KIND	HOW WE USE IT	WHAT WE LEARNED	CURRENT REPO ANCHOR	STATUS
		product and release surface.	operational policy.		
METR, "Measuring AI Ability to Complete Long Tasks" (2025-03-19)	external capability/evaluation reference	Helps explain why the project prefers narrow, rerunnable loops and modest autonomy claims rather than treating all agentic work as equally reliable.	Measuring agent capability by task duration is a useful complement to benchmark-style scores and fits our local-rerun doctrine.	`WHITE_PAPER.md`, `white-paper/RELATED_WORK.md`, `CONFORMANCE_ECONOMICS.md`	`linked`
OpenAI, "PaperBench" (2025-04-02)	external evaluation/reference-design influence	Supports our instinct to decompose complex AI-assisted work into explicit rubrics, gradable subtasks, and reviewer-visible evidence.	Hard agentic work becomes easier to discuss honestly when the grading structure is explicit instead of implied.	`WHITE_PAPER.md`, `white-paper/RELATED_WORK.md`, `CODE_REVIEW_MODEL.md`	`linked`
Galaga gameplay footage, manuals, clips, and extracted artifacts	reference corpus	Grounds Aurora timing, audio, stage cadence, visual comparison, and correspondence work in preserved material.	Manual impressions are useful, but clipped windows, event logs, and aligned audio/visual artifacts make fidelity work reviewable and reusable.	`reference-artifacts/`, `VIDEO_ALIGNMENT_PROGRAM.md`, `CORRESPONDENCE_FRAMEWORK.md`	`linked`
Galaxian gameplay footage and sibling-game source package	reference corpus	Grounds Galaxy Guardians in its own source lineage so the game can become a true sibling application rather than a relabeled Aurora variant.	Ingestion is most valuable when it arrives before design hardens. Second-game credibility depends on game-owned evidence, not borrowed first-game behavior.	`CLASSIC_ARCADE_INGESTION_FRAMEWORK.md`, `APPLICATIONS_ON_PLATINUM.md`, `CONFORMANCE_METRICS_OVERVIEW.md`	`linked`

REFERENCE	KIND	HOW WE USE IT	WHAT WE LEARNED	CURRENT REPO ANCHOR	STATUS
Recovered old-machine source media and representative Neo-Galaga archive	provenance / evidence discipline	Makes source recovery, historical runs, and cited reference media part of the repo-owned evidence program rather than depending on remembered old download paths.	Provenance is stronger when recovered sources have preserved-source lanes, manifests, hashes, and active-doc links instead of only intake notes.	`reference-artifacts/preserved-sources/`, `reference-artifacts/ingestion/downloads-old-all-2026-05-17/`, `WHITE_PAPER.md`	`linked`
Review packet and review-learning ledger model	operational discipline	Makes AI-assisted and fast-moving changes reviewable through durable packets, issue categories, and production dispositions.	Review value compounds when repeated findings become harnesses, release checks, or documented non-goals instead of disappearing into chat.	`CODE_REVIEW_MODEL.md`, `REVIEW_LEARNING_LEDGER.md`	`linked`
Local-first compute doctrine for conformance work	operating doctrine	Pushes repeated measurement into local CPU/browser harnesses while reserving model work for strategy, synthesis, evaluator design, and selected analysis.	The project becomes cheaper and more trustworthy when model assistance leaves behind committed local logic and measurable artifacts.	`CONFORMANCE_ECONOMICS.md`, `PROJECT_STATE_AND_CONFORMANCE_PROGRAM.md`	`linked`

Open Citation Debt

- Recover the earlier standalone Karpathy-related assessment and replace the

current conceptual placeholder with a precise internal or external citation.

- Decide which outside frameworks deserve public-facing mention in the white

paper versus remaining internal working influences.

- Keep a dated related-work refresh cadence so public references are added on

purpose rather than only when remembered opportunistically.

Recurring Project Roles

Updateable definitions for the recurring human, agent, machine, and build-process roles used by the project.

Generated from `white-paper/PROJECT_ROLES.md` during build.

This document is the durable source for recurring human, agent, machine, and build-process roles used by Aurora / Platinum. Update it when a role gains or loses authority, becomes automated, moves to a different tool, or stops being part of the active operating model.

Automation means the role has build, harness, or documentation support. It does not remove human release authority or human review responsibility.

ROLE	DEFINITION	INVOKED OR UTILIZED WHEN	AUTOMATION AND BUILD STATUS	DETAILED DEFINITION OR SOURCE
Manager / consultant	Prioritizes work, interprets evidence, sets stop/go constraints, and asks for cycle handoffs.	Before and after long cycles, when choosing the next quality/security target, and when avoiding rabbit-hole tuning.	Human/session role; outputs are preserved through handoff prompts and plan updates.	`CURRENT_PROJECT_STATE.md`, `LONG_CYCLE_KEEPER_PROCESS.md`, `GO_FORWARD_EXECUTION_PLAN.md`
Developer / execution agent	Implements scoped repo changes, runs checks, preserves unrelated work, commits intentionally, and reports evidence.	Every coding, docs, proof, review, or publish cycle after user or manager direction.	Human/Codex role supported by git, harness checks, build checks, and code-review packet generation.	`AGENTS.md`, `LONG_CYCLE_KEEPER_PROCESS.md`, `MULTI_MACHINE_WORKFLOW.md`
Architect / platform strategist	Converts repeated project pain into reusable mechanisms, schemas, compiler/runtime boundaries, and platform rules.	When a stage-specific fix exposes a systemic gap, or when the project needs reusable game-addition/platform mechanisms.	Mostly human/agent role; architecture outcomes are made checkable through docs, schemas, analyzers, and gates.	`PROJECT_STATE_AND_CONFORMANCE_PROGRAM.md`, `PLATINUM_ARCHITECTURE_OVERVIEW.md`, `CODE_REVIEW_MODEL.md`
Release authority	Controls release-family discipline, hosted `/dev` publish, beta/production authority, and clean-state expectations.	Before branch/release-family decisions, `/dev` publishes, beta/production promotion, and hosted-lane claims.	Partly automated by release checks, publish gates, authority checks, and live verification; beta/production still require explicit user intent.	`MULTI_MACHINE_WORKFLOW.md`, `RELEASE_SCHEDULE_AND_ISSUE_SPINE_2026-06-07.md`, `TESTING_AND_RELEASE_GATES.md`
Parallel worker / iMac M1	Runs separable background work such as Guardians evidence, long persona/watch runs, ingestion cycles, portability checks, docs sweeps, and issue hygiene.	When work can proceed independently without implying release authority.	Machine/human role; artifacts can feed the same build and conformance checks once integrated.	`MULTI_MACHINE_WORKFLOW.md`, `CURRENT_PROJECT_STATE.md`, `PROJECT_WIDE_WORKSTREAM_ALIGNMENT_2026-06-07.md`
Security reviewer	Tracks security findings, severity, release-gate posture, and resolution plans.	Before beta/production, after security-relevant code/data changes, and when a review packet identifies new risk.	Automated by security review and release-gate scripts; generated artifacts feed the human-readable review surface.	`SECURITY_ISSUES_RESOLUTION_PLAN.md`, `security-issues.json`, `tools/build/check-security-release-gate.js`
Code-review gate	Turns changed files, known risks, and security state into a durable review packet with prioritized findings.	Before publish and after material source, docs, harness, or release-surface changes.	Automated by code-review packet and gate scripts, including build/publish review integration.	`CODE_REVIEW_MODEL.md`, `REVIEW_LEARNING_LEDGER.md`, `tools/review/build-code-review-packet.js`, `tools/review/check-code-review-gate.js`

ROLE	DEFINITION	INVOKED OR UTILIZED WHEN	AUTOMATION AND BUILD STATUS	DETAILED DEFINITION OR SOURCE
Conformance evaluator	Measures gameplay, audio, visual, release, confidence, resource economics, and weak-row evidence.	Before accepting keepers, after evidence runs, and before release docs or hosted-lane claims.	Automated by harness analyzers, `npm run harness:measure`, and release conformance dashboard refreshes.	`RELEASE_CONFORMANCE_DASHBOARD.md`, `CONFORMANCE_ECONOMICS.md`, `CONFORMANCE_METRICS_OVERVIEW.md`
Audio review lane	Separates localhost/private reference audio, hosted `/dev` review, public-safe lanes, and foreground-vs-pulse gates.	Before and after audio cue, asset-boundary, or perceived audio-regression work.	Automated by machine audio status, foreground-balance checks, cue-alignment checks, and public artifact boundary tests.	`AUDIO_CONFORMANCE_LAB.md`, `PLATINUM_AUDIO_CONFORMANCE_FRAMEWORK.md`, `tools/dev/private-reference-audio.js`
Player-visible quality reviewer	Decides whether measured movement, audio, visual, or interaction changes actually improve the game for a human player.	During keeper/rejection decisions, proof-to-source lanes, manual review, and beta-relevance discussions.	Part human judgment, part artifact-backed process through before/after captures, contact sheets, scorecards, and strict guardrails.	`LONG_CYCLE_KEEPER_PROCESS.md`, `PLAN.md`, `CHALLENGE_STAGE_CONFORMANCE_ANALYSIS.md`
Documentation generator	Builds hosted guides, white paper pages, slide metadata, PDF metadata, release dashboards, and review surfaces from repo-owned sources.	On normal builds, white-paper review runs, publish preflights, and documentation freshness checks.	Automated by `npm run build`, white-paper review scripts, publish checks, and documentation-freshness gates.	`white-paper/README.md`, `white-paper.json`, `project-guide.json`, `tools/build/build-index.js`

Maintenance Notes

- Keep role authority separate from role automation. A generated check can warn

or block, but it does not create beta/production authority by itself.

- Keep machine roles explicit. The iMac M1 can produce evidence and background

work, but it is not implicit release authority.

- When a new recurring role emerges, add it here first, then link it from the

white paper, project guide, or release docs only if it becomes durable.

- When a build process starts generating or checking a role artifact, update

the automation column and add the script or artifact path.

Illustration Plan

Working list of selected visuals, candidate deeper assets, and the open illustration choices that still need deliberate discussion.

Generated from `white-paper/ILLUSTRATION_PLAN.md` during build.

This document tracks the visuals that support the white paper, the deeper reference materials that should stay nearby in hosted docs, and the places where we still need to choose the most illustrative image or chart deliberately.

Selection Rules

- Prefer visuals that a broad technical or builder reader can interpret quickly.
- Use the main paper for orientation-quality visuals, not for every supporting

artifact.

- Put deeper evidence families in the hosted guides, dashboards, and linked

source documents instead of overloading the main narrative.

- When a visual choice is ambiguous, keep a placeholder in the paper and record

the decision debt here.

Current In-Paper Visuals

WHITE-PAPER SECTION	CURRENT ASSET	WHY IT WORKS
Overview / thesis	`reference-artifacts/diagrams/platinum/platinum-hero.svg`	Gives the paper an immediate project identity and platform-level frame.
Thesis	`export.mov.png`	Shows that the evidence and release program serve a real playable artifact.
Program snapshot	`reference-artifacts/diagrams/platinum/aurora-pack-card.svg`	Makes Aurora legible as an application on the platform.
Program snapshot	`reference-artifacts/diagrams/platinum/galaxy-guardians-pack-card.svg`	Shows second-game identity without requiring a long explanation.
Five-layer operating model	`reference-artifacts/diagrams/platinum/platinum-platform-stack.svg`	Reinforces platform-versus-application separation.
Five-layer operating model	`reference-artifacts/diagrams/platinum/platinum-pack-separation.svg`	Helps explain ownership boundaries at a glance.
Ingestion strategy	`reference-artifacts/analyses/galaga-stage-opening-timing/2026-04-12-main-a777fba/opening-contact-tight.png`	Makes ingestion concrete through a visible reference-study artifact.
Ingestion strategy	`reference-artifacts/analyses/galaxian-reference/matt-hawkins-arcade-intro/frames/contact-sheet-reference-window.jpg`	Supports the claim that Galaxy Guardians is grounded in its own source family.
Harnessing and conformance	`reference-artifacts/analyses/conformance-economics/2026-05-14-1c788342/score-trends.svg`	Turns progress into an at-a-glance measurable story.
Harnessing and conformance	`reference-artifacts/analyses/persona-performance-distribution/performance-lines.svg`	Shows that quality is evaluated across viewpoints, not through one metric alone.
Release and economics	`reference-artifacts/analyses/conformance-economics/2026-05-14-1c788342/compute-minutes-by-resource.svg`	Makes local-first measurement strategy visible.
Release and economics	`reference-artifacts/analyses/conformance-economics/2026-05-14-1c788342/cost-per-positive-score-point.svg`	Connects release ambition to investment discipline.

Deeper Supporting Visuals

These are better linked from hosted guides or follow-on detail pages than pushed directly into the main narrative unless a specific section needs them.

- `reference-artifacts/analyses/conformance-economics/2026-05-14-1c788342/largest-score-deltas.svg`
- `reference-artifacts/analyses/conformance-economics/2026-05-14-1c788342/gpu-equivalent-use-by-purpose.svg`
- `reference-artifacts/analyses/conformance-economics/2026-05-14-1c788342/cpu-use-by-purpose.svg`
- `reference-artifacts/analyses/conformance-economics/2026-05-14-1c788342/gameplay-improvement-by-project-part.svg`
- hosted `conformance-dashboard.html`
- hosted `release-dashboard.html`
- hosted `project-guide.html`

- hosted `public-project-page.html`

Open Illustration TODOs

Release-history progression strip

- White-paper placeholder exists.
- Best candidate still needs discussion:
- `1.0.0` public game surface
- `1.2.0` Platinum framing surface
- `1.4.0` multi-game and conformance surface
- Decision question: should the strip emphasize gameplay evolution, platform

architecture, or public release/documentation maturity?

Ingestion in action

- White-paper placeholder exists.
- Best candidate still needs discussion:
- contact sheet only
- waveform plus contact sheet pair
- raw source clip to structured artifact pipeline graphic
- Decision question: what most clearly teaches the ingestion idea to a

non-expert reader in one glance?

Evidence loop case study

- White-paper placeholder exists.
- Best candidate still needs discussion:
- question -> artifact -> harness -> result graphic
- screenshot pair showing before and after measured correction
- release-note excerpt plus chart plus artifact collage
- Decision question: which single case study best shows how AI-assisted work is

kept honest by rerunnable evidence?

Milestone gallery

- White-paper placeholder exists.
- Best candidate still needs discussion:
- one screenshot per milestone release
- one diagram per maturity phase
- one mixed gallery of game, platform, and dashboard surfaces
- Decision question: should the gallery feel product-led, architecture-led, or

release-led?

Likely Next Additions

- Add one screenshot from the hosted conformance dashboard.
- Add one screenshot from the hosted release dashboard.
- Add one architectural illustration that shows ingestion, games, Platinum, and

release lanes on a single page.

- Add one progression visual based on preserved historical release surfaces.

Related Work Log

Dated log of external references, why they matter here, and when they should be refreshed.

Generated from `white-paper/RELATED_WORK.md` during build.

This log tracks external work worth preserving alongside the white paper.

It is deliberately narrower than a literature survey. The goal is to keep a small, high-signal set of references that help explain why this project looks the way it does.

Working Rule

- Prefer focused, occasional searches over bulk accumulation.
- Prefer primary or close-to-primary sources when the topic is methodological.
- Record why the work matters here, not only that it exists.
- Update the main white paper only with the most relevant current items; keep

the longer memory here.

Current Seed Set

DATE ADDED	WORK	RELEVANCE HERE	LINK
`2026-05-16`	Anthropic, *Building effective agents* (`2024-12-19`)	Supports our preference for simple loops, visible evaluator structure, and avoiding ornamental agent complexity.	Source
`2026-05-16`	Anthropic, *Writing effective tools for agents - with agents* (`2025-09-11`)	Closely matches our view that harnesses, scripts, and release tools are explicit contracts between model assistance and deterministic behavior.	Source
`2026-05-16`	Anthropic, *Demystifying evals for AI agents* (`2026-01-09`)	Reinforces our investment in evals, reruns, transcripts, graders, and narrow measurable questions.	Source
`2026-05-16`	Anthropic, *Trustworthy agents in practice* (`2026-04-09`)	Supports the idea that controls, guardrails, human review, and operator-visible release process are product concerns, not only policy concerns.	Source
`2026-05-17`	METR, *Measuring AI Ability to Complete Long Tasks* (`2025-03-19`)	Useful because it frames agent capability in terms of task duration and reliability, which helps explain why this project prefers small, reviewer-visible loops and local reruns over stronger autonomy claims.	Source
`2026-05-17`	OpenAI, *PaperBench* (`2025-04-02`)	Useful because it shows how hard agentic work becomes more reviewable when decomposed into explicit rubrics and many gradable subtasks, which is close to how our own conformance categories make progress inspectable.	Source

Refresh Triggers

- major white-paper revision
- major release-policy or conformance-process revision
- meaningful new methodological influence worth naming publicly
- significant change in the broader agent/evals/tooling conversation that bears

directly on this project

Reviewer Checklist

Release-minded checklist for reading the white paper critically across narrative, HTML, PDF, and related-work quality.

Generated from `white-paper/REVIEWER_CHECKLIST.md` during build.

This checklist is meant to make the reviewer mentality explicit.

The white paper is part of the release surface. It should be reviewed with the same seriousness as other user-visible documentation and release artifacts.

Core Stance

- Read as a reviewer, not as the author.
- Prefer tighter and clearer over longer and more complete.
- Challenge repetition, overclaiming, weak evidence links, and vague diagrams.
- Treat HTML presentation, PDF presentation, and narrative quality as one

release concern.

Narrative Review

- Is the thesis clear in the first page?
- Does each section earn its length?
- Are repeated ideas tightened rather than restated?
- Are claims linked to evidence, docs, or clearly named future work?
- Does the paper stay readable for a broad technical or builder audience?

HTML Review

- Does the hosted page read well on desktop and mobile?
- Does the hero lead the page clearly before the supporting surfaces?
- Is the overview deck linked next to the white paper from both the

white-paper page and the public project page?

- Are diagrams, tables, and screenshots legible without awkward empty runs?
- Do callouts render intentionally instead of leaking raw Markdown markers?
- Are deeper-detail links obvious without overwhelming the page?

Slide Overview Review

- Does the overview deck tell the project story without requiring repo context?
- Does it reflect the current white-paper version and updated date?
- Are current conformance, ingestion, economics, review, and go-forward claims

consistent with the white paper and dashboard artifacts?

- Are the source-artifact links useful for a reader who wants to drill down?

PDF Review

- Does the PDF show the white-paper version and updated date clearly?
- Are page breaks reasonable around diagrams, images, tables, and callouts?
- Are any sections visually repetitive or too sparse when printed?
- Is the print styling readable and professional rather than a dark-theme dump?
- Do diagrams and screenshots stay large enough to be useful?

Release Review

- Run `npm run white-paper:review` for the active dev draft.
- Run `npm run white-paper:review:beta` or

`npm run white-paper:review:production` before publishing those lanes.

- Treat the review command as responsible for refreshing the code-review packet

and passing the review gate, not only for rendering a PDF.

- Confirm the review pass also checks preserved-source integrity so active docs

and timing/audio reference work do not drift back to stale machine paths.

- Commit the refreshed review packet before the clean-tree publish step.
- Confirm the generated PDF metadata matches the white-paper version/date.
- Confirm the generated slide metadata matches the white-paper version/date.
- Verify the live lane carries `white-paper.html`, `white-paper.pdf`,

`project-overview-slides.html`, and `project-overview-slides.json` after publish.

Related Work Review

- If outside work materially shaped the draft, is it logged?
- If a web refresh was done, are the selected items high-signal and relevant?
- Is the related-work section in the main paper still concise?
- Does the deeper log explain relevance instead of just listing links?

Review Cadence

Small recurring rhythm for when to run the white-paper review spine and what it should catch beyond formatting alone.

Generated from `white-paper/REVIEW_CADENCE.md` during build.

This note turns the reviewer mentality into a small recurring operating rhythm instead of leaving it as a good intention.

Default Cadence

On every meaningful white-paper or preserved-source change

- run `npm run white-paper:review`
- treat the result as a release-surface check, not just a PDF render

Before `/beta` or `/production` publication

- run `npm run white-paper:review:beta` or

`npm run white-paper:review:production`

- verify the lane carries `white-paper.html`, `white-paper.pdf`,

`project-overview-slides.html`, and `project-overview-slides.json`

During active fast-moving project periods

- do at least one focused review pass per week on:
  - white-paper clarity
  - overview deck consistency
  - HTML/PDF presentation
  - preserved-source integrity
  - stale path drift
  - related-work freshness

On major narrative or methodology shifts

- consider cutting a new white-paper snapshot under `white-paper/releases/`
- refresh `white-paper/project-overview-slides.json` in the same pass
- refresh the citation ledger and related-work log in the same pass

What The Review Pass Should Catch

- repeated or softened claims that can be tightened
- diagrams, tables, or screenshots that create awkward whitespace
- PDF page breaks that weaken readability
- stale machine-specific source paths in active evidence docs
- preserved-source drift that would make analysis less rerunnable
- related-work references that have gone stale or no longer feel central

Current Review Spine

- `npm run white-paper:review`
  - refresh the dev-lane PDF
  - refresh and check the generated overview deck

- refresh the code-review packet
- pass the code-review gate
- verify the preserved-source integrity check
- verify the hosted white-paper presentation check

White Paper Working Area

How the project area is organized and when new narrative snapshots should be cut.

Generated from `white-paper/README.md` during build.

This directory holds the durable support files for the project white paper.

Purpose

The white paper is meant to be both:

- a maintained promotional explanation of what Platinum, Aurora, Galaxy

Guardians, ingestion, harnessing, conformance, and release discipline are trying to achieve

- a durable internal reminder of what the project is doing, why it is doing it,

and how the method evolves over time

Structure

- `../WHITE_PAPER.md`
  - current living draft
- `project-overview-slides.json`
  - durable source for the generated public overview deck
- `PROJECT_ROLES.md`
  - durable source for recurring project roles, invocation timing, and role

automation/build status

- `CITATION_LEDGER.md`
  - maintained list of outside ideas, source families, and project learnings
- `ILLUSTRATION_PLAN.md`
  - selected visuals, candidate deeper assets, and open illustration decisions
- `RELATED_WORK.md`
  - dated log of external/public work, why it matters here, and where it fits
- `REVIEWER_CHECKLIST.md`
  - release-minded checklist for HTML, PDF, narrative, and citation review
- `REVIEW_CADENCE.md`
  - the small recurring rhythm for when and how the white-paper review pass

should be run

- `releases/<date>-v<version>/WHITE_PAPER.md`
  - exact snapshot of the white paper for a given release
- `releases/<date>-v<version>/WHITE_PAPER.pdf`
  - printable/exportable snapshot for that white-paper release when available
- `releases/<date>-v<version>/WHITE_PAPER_PDF_METADATA.json`
  - version/date/build metadata for the release PDF snapshot when available
- `releases/<date>-v<version>/RELEASE_NOTES.md`
  - short explanation of what changed in that white paper release

Maintenance Rules

- Update `../WHITE_PAPER.md` directly while shaping the next narrative draft.
- Cut a new snapshot when the white paper changes in a strategically meaningful way.
- Prefer one snapshot per meaningful narrative release rather than one snapshot per tiny wording edit.
- When cutting a snapshot, preserve the Markdown, the generated PDF, and the PDF metadata together whenever the release PDF exists.
- Keep the citation ledger current whenever outside work, source material, or methodological influences materially shape the paper.
- Keep the illustration plan current whenever a new screenshot, chart, diagram, or hosted-detail surface becomes important to the white-paper story.
- Keep the related-work log current when a focused web refresh produces a source worth preserving for future readers.
- Keep the recurring project roles current when a human, agent, machine, or build-process role changes authority, invocation timing, or automation status.
- Treat HTML and PDF presentation review as part of release quality, not only as a late formatting pass.

Maintained Release Surfaces

- `dist/<lane>/white-paper.html`
  - hosted readable white-paper surface for a release lane
- `dist/<lane>/white-paper.pdf`
  - printable/exportable white-paper surface for a release lane
- `dist/<lane>/white-paper-pdf.json`
  - version, date-updated, lane, and build metadata for the generated PDF
- `dist/<lane>/project-overview-slides.html`

- hosted project overview deck, linked next to the white paper

- `dist/<lane>/project-overview-slides.json`
  - slide metadata and source trace for release review and public inspection

Suggested Review Commands

- `npm run white-paper:review`
  - refresh the dev-lane PDF, reviewer packet, review gate,

preserved-source-integrity check, project overview deck, and presentation checks together

- `npm run white-paper:review:beta`
  - run the same white-paper review spine for the beta lane

- `npm run white-paper:review:production`
  - run the same white-paper review spine for the production lane

Suggested White Paper Release Triggers

- major public release family change
- major conformance-program reframing
- significant Galaxy Guardians maturity change
- meaningful new evidence or economics story
- recovery of an important prior assessment or external citation

Current Snapshot

The first seeded snapshot is:

- `releases/2026-05-16-v0.1.0/`

This establishes the initial narrative baseline, the release structure, and the citation ledger.

White Paper v0.1.0 Release Notes

Release note for the first seeded white-paper snapshot.

Generated from `white-paper/releases/2026-05-16-v0.1.0/RELEASE_NOTES.md` during build.

Release: `2026-05-16-v0.1.0` Date: `2026-05-16` Type: foundational narrative release

Summary

This is the first seeded release of the project white paper.

It establishes:

- a single maintained narrative for Platinum, Aurora, Galaxy Guardians,

ingestion, harnessing, conformance, release discipline, and AI-assisted engineering

- a dedicated citation ledger
- a release structure for preserving historical white paper versions over time

What This Release Adds

- a preamble explaining the broader project claim
- a section-by-section first draft of the white paper
- diagrams and visual anchors for the operating model
- an initial timeline of major project phases
- a clear note that the earlier Karpathy-style assessment still needs to be

recovered and linked directly

Recommended Next Pass

- tighten the audience voice depending on whether the next target is public

promotion, collaborator onboarding, or AI-method storytelling

- add selected screenshots or dashboard graphics
- recover the missing prior citation material
- revise the paper alongside the next strategic release or conformance shift