ARCHITECTURE VISION BRIEF

THE GREAT NORTH

Mainely Code Providence Architecture

SPECIALIZED PRODUCTS. GOVERNED EXECUTION. PROOF END TO END.

The Great North
The complete Mainely Code ecosystem.

Providence
The governed architecture connecting cognition, products, infrastructure, and proof.

THE ECOSYSTEM

A connected system - not a single app.

The Great North is Mainely Code's product and platform ecosystem. It brings specialized experiences together through a governed architecture while preserving a clear job, authority, and success standard for each product.

The Great North

The public umbrella for the complete Mainely Code ecosystem: cognition, cloud experience, design, software engineering, first-party inference, trusted compute, identity, and proof.

Providence

The formal architecture connecting those systems through explicit authority, versioned contracts, governed execution, and evidence across every handoff.

The Great North runs on Providence.

Providence means foresight, responsible stewardship, and coordinated guidance - translated into practical software boundaries and proof.

Specialized

Each product is built for a distinct customer job.

Governed

Every authority has a defined right to act and a defined limit.

Verifiable

Important outcomes carry evidence, not just confident language.

THE PRODUCT FAMILY

Four experiences. One coherent direction.

Mainely Code Northstar

Flagship cognitive platform

Coordinates specialist reasoning, policy, budgets, provenance, and product-family routing into one inspectable operating layer.

Project Sasquatch

Cloud command center

Delivers broad Mainely Code AI capability through cloud conversations, routing, account controls, and operator-facing workflows.

Mainely Code Buildroom

Proof-driven software engineer

Turns bounded engineering requests into candidate changes, verification, proof bundles, rollback paths, and verified PR/apply proposals.

Orb-Weaver

Local-first UI/UX design loom

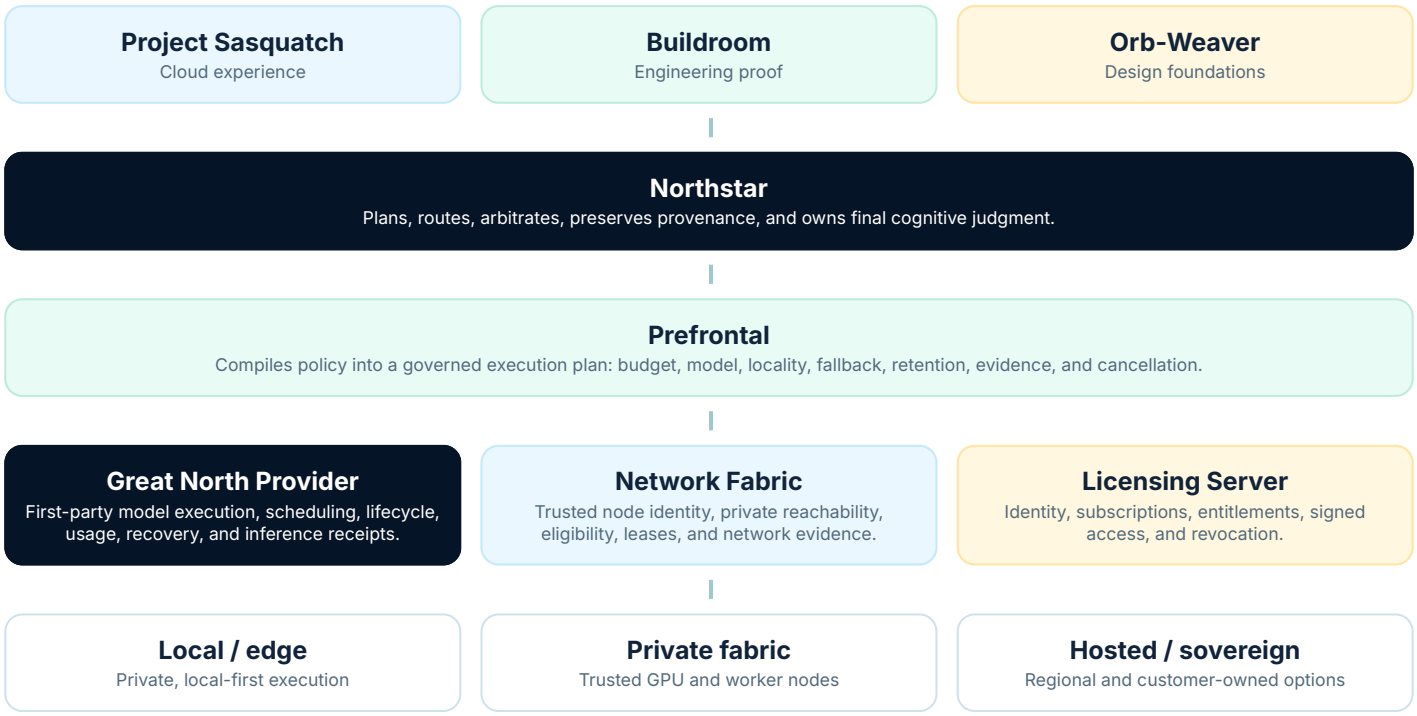
Turns prompts, sketches, screenshots, and product-flow ideas into reviewable maps, components, design tokens, previews, and editable interface foundations.

The handoff matters: Sasquatch clarifies and routes. Orb-Weaver designs. Buildroom implements and proves. Northstar coordinates the larger plan.

THE ARCHITECTURE

Providence at a glance

A clear chain connects customer intent to governed execution without allowing any one component to become an unchecked authority.



RESPONSIBILITY BY DESIGN

One concern. One authority.

Providence is designed to make powerful AI systems easier to understand, control, and audit by separating decisions that are often blurred together.

Northstar decides

Goals, decomposition, specialist coordination, budgets, provenance, and final cognitive route.

Prefrontal permits

Policy, locality, fallback, tools, retention, evidence, cancellation, and risk constraints.

Great North Provider executes

Exact model release, admission, scheduling, runtime lifecycle, streaming, cancellation, usage, and recovery.

Network Fabric establishes trust

Node identity, private reachability, eligibility, leases, network state, and infrastructure evidence.

Licensing Server authorizes access

Customer identity, subscription, entitlement, product audience, and revocation.

Products accept outcomes

Buildroom verifies engineering. Orb-Weaver validates design output. Sasquatch owns the cloud journey.

No component gets to grade its own homework. The system that executes inference cannot silently rewrite policy, invent network trust, grant access, or declare a product outcome successful.

FIRST-PARTY INFERENCE

Great North Provider

Great North Provider is the first-party inference system being developed for the Great North: a runtime, scheduler, release lifecycle, trusted-node layer, usage ledger, and evidence system - not simply another endpoint.

Exact model identity

Certified requests bind to an immutable release that includes the weights, tokenizer, template, configuration, lineage, license state, and approved runtime profile.

Governed execution plans

Policy can define the allowed model, deadline, budget, locality, fallback, cache, retention, evidence, and cancellation behavior before execution begins.

Inference receipts

A receipt can record what ran, where it ran, how long it took, what resources were used, whether fallback occurred, and how the result binds to the plan.

Recoverable operations

Lifecycle, queue, cancellation, restart, unknown-outcome, rollback, and quarantine behavior are treated as product features rather than operational afterthoughts.

> Direction from Northstar. Execution through Great North Provider. Proof throughout Providence.

Great North Provider is under active development. Specific runtime, hardware, deployment, and performance claims will be published only for certified release paths.

DOGFOODING

We intend to use our own system first.

Mainely Code's products provide demanding real-world workloads for Providence. That makes internal use part of the qualification system, not a marketing demonstration.

1 - Plan
Northstar defines the route.

→

2 - Govern
Prefrontal compiles the limits.

→

3 - Execute
Provider runs the exact release.

→

4 - Prove
A receipt records the inference.

→

5 - Verify
Buildroom or another product accepts or rejects the outcome.

Why Buildroom comes first

Software engineering exposes weak cancellation, stale model identity, poor structured output, bad context selection, runaway retries, false completion, and incomplete recovery faster than a chat demo.

What counts as healthy

Not "the model returned text." A healthy path is exact, bounded, recoverable, receipt-backed, and accepted by the consuming product's real verification contract.

Proof is the feedback loop. Verified outcomes can improve evaluation, routing, release promotion, hardware profiles, and cost planning without treating uncontrolled customer content as training material.

DEPLOYMENT CHOICE

Built for more than one place.

Providence is designed so the same authority and evidence contract can extend from a local machine to a private fabric or a governed hosted region.

Local / edge

Private, local-first execution for products and workloads that benefit from endpoint control, lower latency, or limited connectivity.

Private fabric

Trusted customer, office, lab, or private-cloud nodes connected through signed identity, policy, leases, and evidence.

Hosted / sovereign

Regional, dedicated, or customer-owned deployment paths designed around isolation, residency, audit, and operational control.

Portable product contract

The product should not change its definition of success because the model ran in a different location.

No silent relocation

Residency and locality belong in the governed plan. A cheaper or available region cannot silently override them.

Customer choice is part of the architecture: local where practical, private where required, hosted where useful - with the same expectation of exact identity and evidence.

WHY IT MATTERS

Control without mystery.

Privacy and locality

Keep sensitive work local or private when the workload and product allow it. Make retention and data movement explicit.

Portability

Separate product contracts from one vendor, one model, one cloud, or one hardware profile.

Cost visibility

Bind estimates, budgets, retries, usage, and actual outcomes instead of hiding expensive loops behind vague credits.

Operational resilience

Treat cancellation, restart, unknown outcomes, rollback, and degraded modes as designed behavior.

Evidence

Record what ran and let the consuming product determine whether the result actually solved the customer's problem.

Responsible authority

Keep cognition, policy, compute trust, access, inference, and product acceptance separate enough to challenge one another.

Powerful systems deserve visible boundaries.

Providence is the commitment to make those boundaries understandable, enforceable, and provable.

CURRENT DIRECTION

Ambitious architecture. Honest release posture.

The Great North is an active Mainely Code program. Product and platform capabilities are being built and qualified in stages rather than represented as finished before the evidence exists.

Now

Source and architecture lock

Product separation, authority boundaries, provider direction, proof rules, and canonical naming.

Next

First-party execution proof

Exact model release, governed local execution, receipt, Buildroom shadow/canary, failure and rollback evidence.

Then

Trusted private mesh

Signed remote nodes, private artifact delivery, distributed scheduling, resilience, and deployment certification.

Availability statement: Individual Mainely Code products and capabilities may be in development, preview, beta, certification, or general availability at different times. Product-specific release pages remain the source of truth.

What we will publish

Named release notes, supported hardware/model profiles, security and deployment boundaries, benchmark methodology, known limitations, and proof-backed claims.

What we will not publish as fact

Unsupported superiority claims, vague “fully autonomous” promises, silent fallback behavior, or success claims that have not survived product-specific verification.

THE GREAT NORTH

A system built around responsibility.

Mainely Code is building a family in which design, cognition, software engineering, inference, infrastructure, and access can work together without becoming one unaccountable black box.

THE GREAT NORTH

A MAINELY CODE PROVIDENCE ARCHITECTURE

The ecosystem

The Great North is the connected Mainely Code world.

The architecture

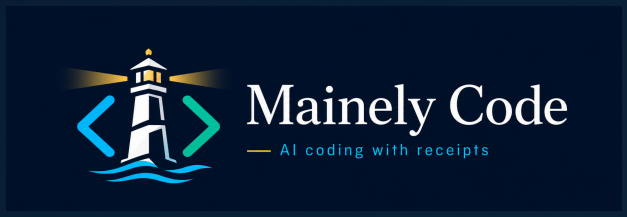
Providence defines how authority, trust, execution, and proof work together.

The inference plane

Great North Provider is the first-party system for governed model execution and evidence.

The standard

Specialized products. Governed execution. Proof end to end.



MAINELYCODE.COM