Dweve
Weaving Intelligence

ENGLISH EDITION · JULY 2026

MARKETING BROCHURE · 16 PAGES

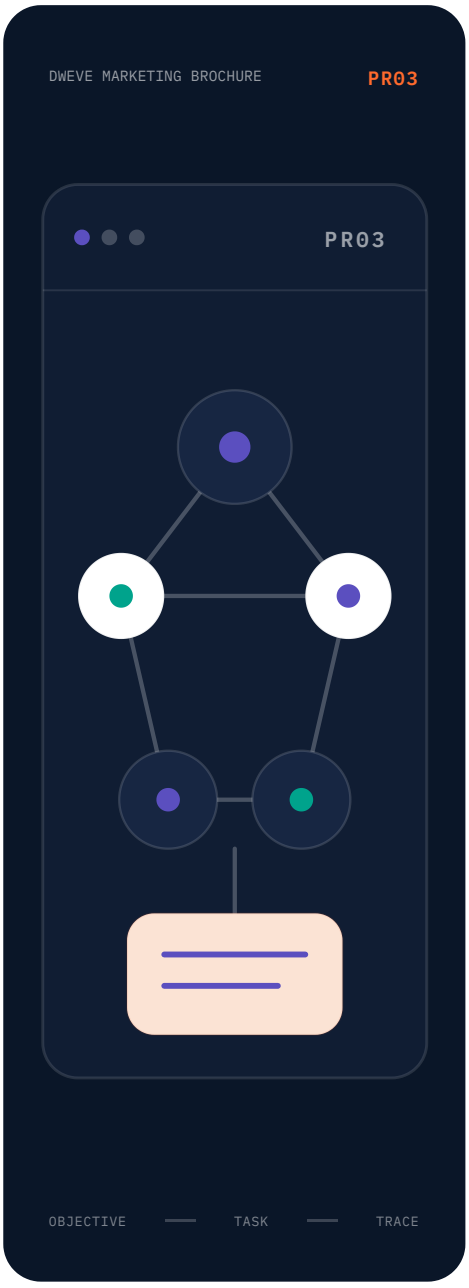
Nexus: the product brochure

A standalone visual introduction to Nexus, the job it owns, how it works, where it fits, and how to prove it.

VISUAL BRIEFING

PRODUCT LEADERS, BUYERS, ARCHITECTS, DELIVERY TEAMS, AND TECHNICAL EVALUATORS

ENGLISH



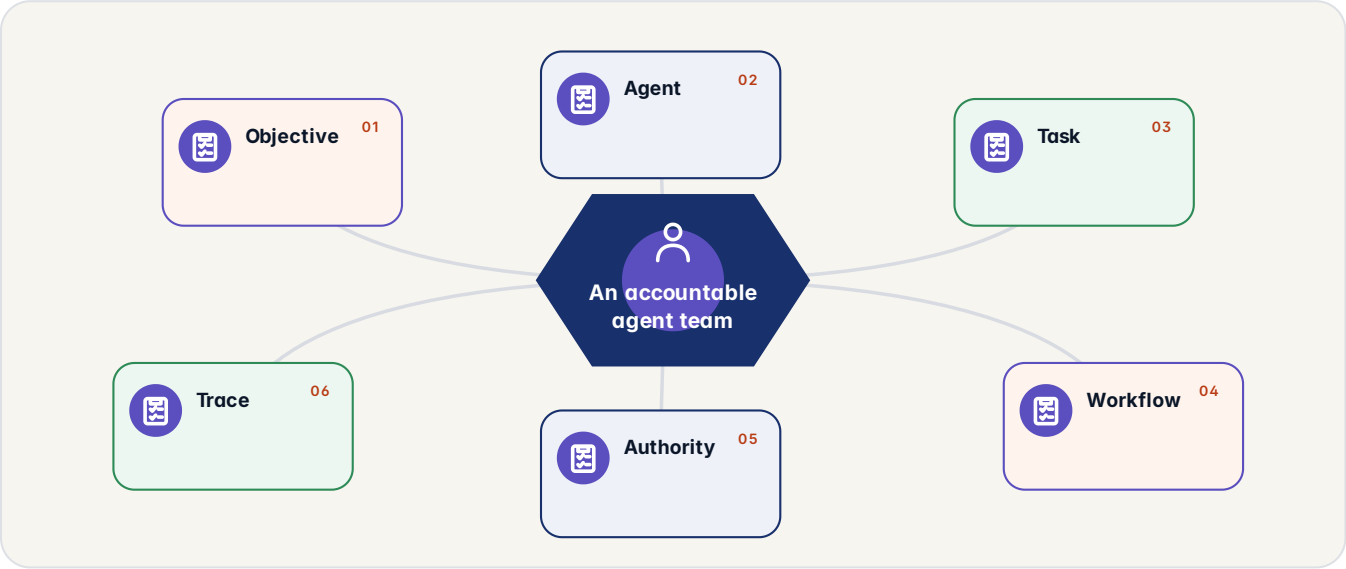
Decide whether Nexus belongs in the selected workflow.

PR03

THE PROPOSITION

An accountable organisation around an objective

Nexus is the multi-agent organisation layer inside Fabric. It is what allows that workspace to call on several specialised capabilities without turning your work into several disconnected applications.



How to read this visual: follow an accountable agent team through the proposition.

DECISION

Read the promise against the operating reality

A model can answer. An agent can act. Neither becomes an organisation simply because more of them are connected. Dweve Nexus assembles specialised agents, models, tools, memory, policies, workflows, and human authority around an objective, then runs them as one accountable operation. Every participant receives a role. Every task has an owner. Every handover remains attached to the work. Every action stays inside its authority. Every contribution returns to one outcome. The task does not enter another chat. It becomes an organisation.

Traditional organisations maintain permanent structures and then route every new problem through them. Nexus can do the reverse. It starts with the objective, decomposes it into responsibilities, dependencies, decision points, parallel work, quality checks, and approval boundaries, then determines which capabilities the work requires and forms the smallest useful organisation around it.

- Objective
- Agent
- Task
- Workflow

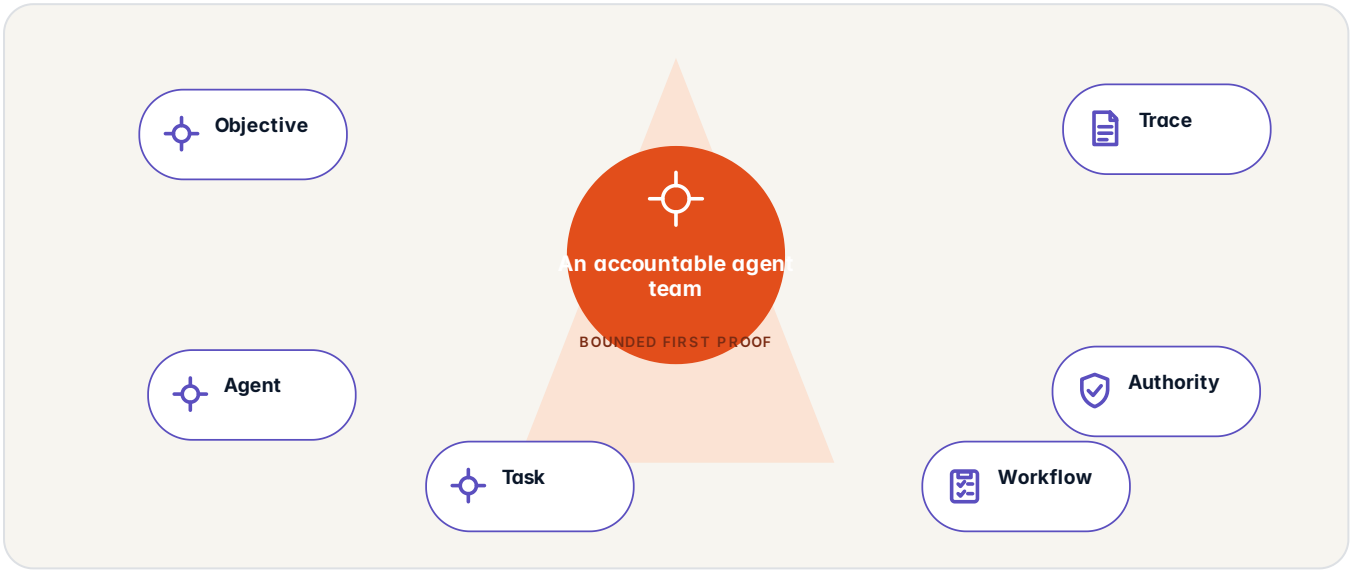
QUESTION TO CARRY

Which part of this proposition changes the outcome people receive today?

USE AND FIT

Where Nexus can earn its place

Start with responsibilities that have a real owner, useful sources, and a consequence worth improving.



How to read this visual: follow an accountable agent team through use and fit.

MECHANISM

Separate useful scope from attractive distraction

Nexus can learn which specialists and ways of working succeed for different kinds of tasks. The team can improve without becoming unpredictable and without giving itself new authority. Better experience changes how the organisation is formed. It does not change who is in charge. A specialist that repeatedly handles a certain job well can become more likely to receive similar work. A successful sequence can become a reusable workflow. A useful review step can become part of the procedure.

Nexus does not treat memory as one endless transcript. It remembers different things for different reasons. A specialist receives the memory relevant to its role. The useful context follows the work. Everything else stays out of the way. Working memory keeps track of the task happening now. Episodic memory remembers what happened in earlier conversations and cases. Semantic memory holds facts, documents, relationships, and knowledge you want to keep. Procedural memory remembers ways of working that you use repeatedly. The agent checking an expense does not need every family conversation.

- Objective
- Agent
- Task
- Workflow

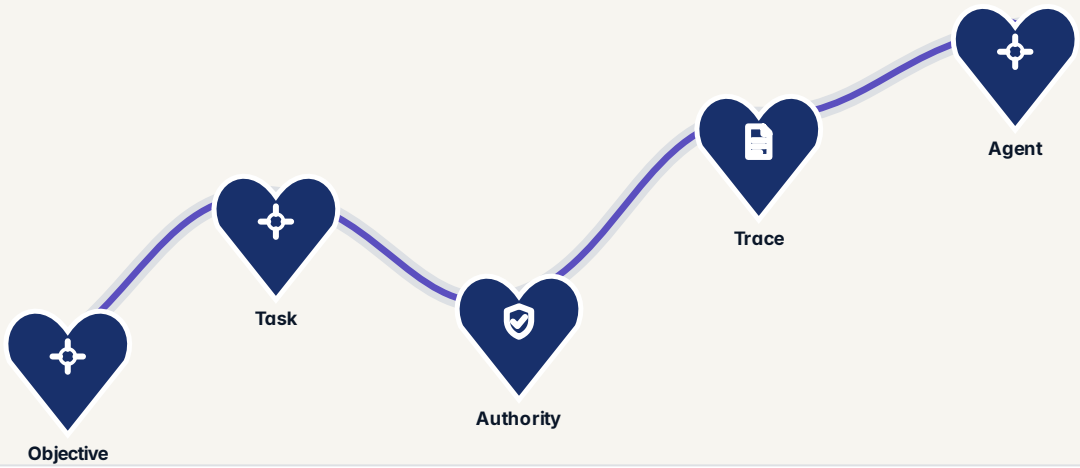
QUESTION TO CARRY

Where does this earn a place, and where should it deliberately not be used?

THE PATH

How Nexus moves from question to decision

A focused engagement should make the next decision easier, not create a larger promise.



How to read this visual: follow an accountable agent team through the path.

WHY THIS SHAPE WORKS

The workflow, operating boundary, evidence, and decision criteria are considered together from the start.

DECISION

Make every stage earn the next one

A prompt loop is enough to demonstrate tool use. It is not enough to run an organisation. Production multi-agent systems need agent identity, capability discovery, task state, workflow topology, collaboration structures, authority, memory, resource budgets, service levels, recovery, observability, and reproducible routing. Dweve Nexus is a multi-agent runtime written in Rust, with bindings for Python, Go, and Node.js. Models remain components. Agents remain bounded workers. Tasks remain state machines. Workflows remain graphs. Authority remains enforceable. Every routing decision produces a trace.

Nexus records the observable path behind the result: the original objective, the decomposition, the agents considered, the routing decision, the roles assigned, the policies evaluated, the memory and sources retrieved, the tools invoked, the approvals received, the failures encountered, the recovery path taken, the outputs produced, and the final outcome. This is not a claim that Nexus exposes a model's private chain of thought. It records the operational decisions made by the agent system. A reviewer can reconstruct the work without treating generated prose as evidence.

- Objective
- Task
- Authority
- Trace

QUESTION TO CARRY

What evidence must exist before the scope is allowed to grow?

THE VALUE PICTURE

Nexus seen from three responsibilities

Good AI work must make sense to the person using it, the team operating it, and the people reviewing it.

How to read this visual: follow an accountable agent team through the value picture.

PRIMARY READER

Product leaders, buyers, architects, delivery teams, and technical evaluators

The first conversation.

DECISION SUPPORTED

Decide whether Nexus belongs in the selected workflow.

The next useful step.

MECHANISM

Read the same outcome from several responsibilities

An organisation must remember facts, cases, procedures, responsibilities, and outcomes. Nexus separates memory by purpose. Working memory holds the active task, current constraints, intermediate results, and unresolved decisions. Episodic memory preserves what happened in previous cases and how those cases ended.

Bring Nexus an objective that currently crosses people, systems, rules, and approvals. See how the work is decomposed, how the organisation forms, how authority is assigned, how failure is contained, and how every contribution returns to one accountable outcome.

- Objective
- Task
- Authority
- Agent

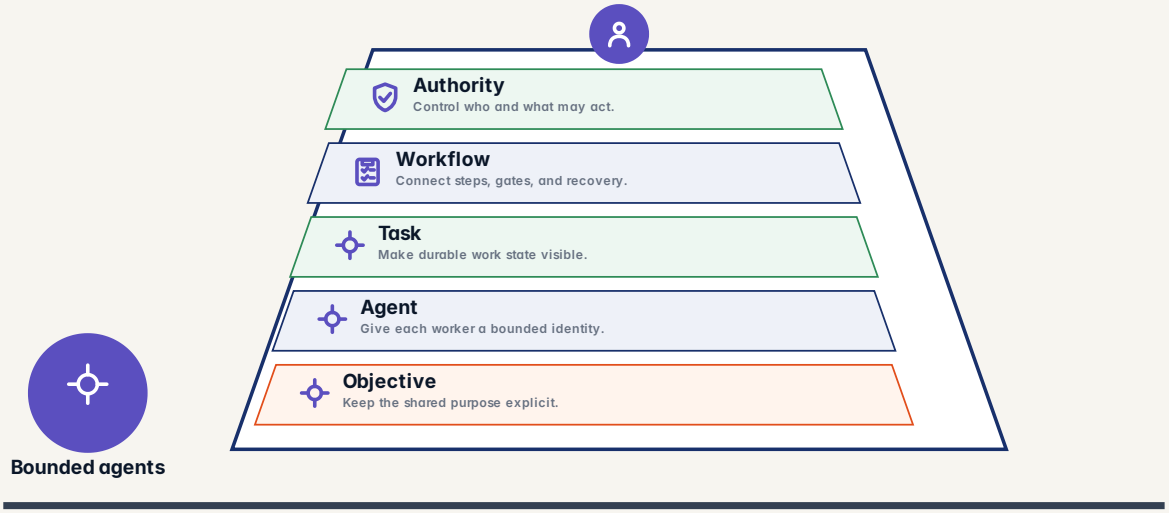
QUESTION TO CARRY

Can the same result satisfy its user, operator, and reviewer?

THE OPERATING MODEL

The control model around Nexus

A credible proposition connects the visible result to the less visible responsibilities underneath it.



How to read this visual: follow an accountable agent team through the operating model.

DECISION

Name the controls behind the simple interface

An organisation formed for one task can become a repeatable operating model. Successful roles can become templates. Reliable handovers can become workflow structures. Recurring approval points can become policy gates. Useful procedures can become procedural memory. Demonstrated specialisation can improve future staffing. The next organisation does not need to rediscover everything from nothing. It can inherit what worked while still being assembled around the new objective. The inheritance is selective: the next objective takes the role templates, workflow structures, and policy gates that fit, and leaves the rest on the shelf. Nothing is cloned wholesale, and nothing useful is thrown away.

External systems remain external: a model provider can change, a website can return new information, a business record can be updated, an API can behave differently. Nexus records those boundaries and the values that crossed them. A replay can use captured outputs to reconstruct the original operation, or call the dependency again and show where divergence entered. The promise is not that the world never changes.

- Objective
- Agent
- Task
- Workflow

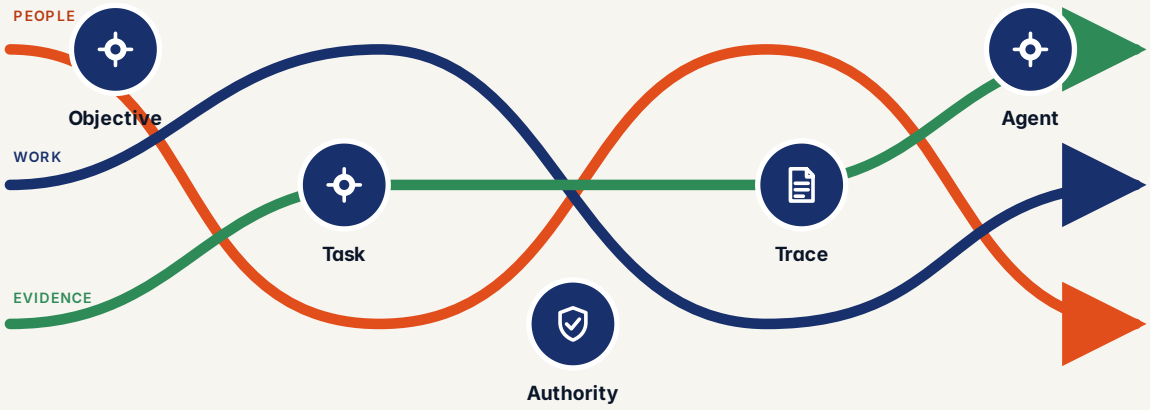
QUESTION TO CARRY

Who owns each control when the interface appears simple?

THE RESPONSIBILITY FLOW

People, Nexus, and evidence move together

The work becomes easier to govern when each stage leaves a clear hand-off.



How to read this visual: follow an accountable agent team through the responsibility flow.

MECHANISM

Keep every hand-off attached to an owner

Semantic memory holds durable facts, concepts, relationships, and policy knowledge. Procedural memory preserves ways of working that can be reused. Each agent receives the memory relevant to its role and current responsibility, not unrestricted access to the organisation's entire history. When the temporary team dissolves, the organisation does not forget what it learned. Underneath, that memory is binary and deterministic, held to the same arithmetic discipline as the routing.

Ask once. Let Nexus form the team, divide the work, bring the pieces together, and return one result. The organisation works for you. The authority remains with you.

- Objective
- Task
- Authority
- Trace

QUESTION TO CARRY

Is every hand-off visible, bounded, and assigned to an owner?

SOURCE TO RESULT

Nexus: from authoritative source to result

The work stays explainable when source identity, transformation, decision, and hand-off remain visible between systems.

How to read this visual: follow an accountable agent team through source to result.

DECISION

Keep the basis visible as the work changes form

But as the number of agents grows, so does the operational ambiguity: who owns the complete objective, which agent may commit the organisation, which source is authoritative, what happens when two agents disagree, who notices that a handover failed, which memory belongs to the current case, and where responsibility returns when the work is finished. Nexus supplies the structure that turns individual capabilities into a functioning operation.

External dependencies remain possible sources of change: a hosted model may produce a different output, a tool may return new data, a web resource may change, a business system may be mutated between executions. Nexus records those calls, their position in the run, the version information available, the inputs sent, and the outputs returned. Captured replay reuses recorded dependency outputs to reconstruct the original controlled execution; live replay calls the dependency again and compares. When the operation diverges, the diff can identify the layer and external boundary at which the change entered.

- Identify
- Prepare
- Apply
- Review

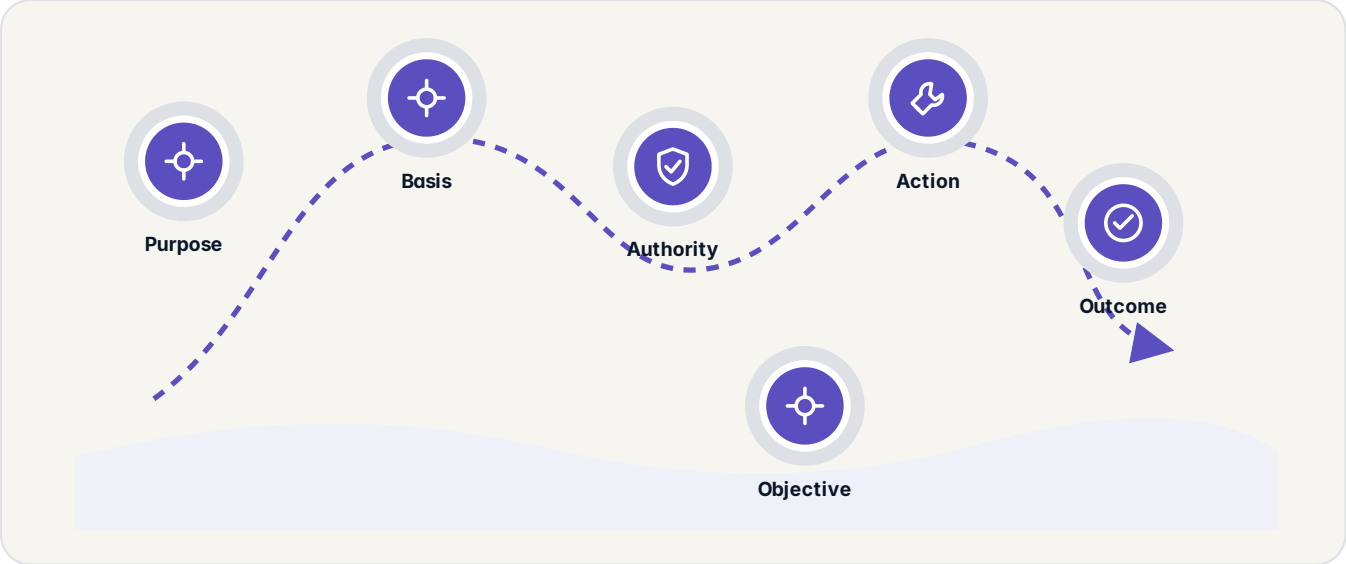
QUESTION TO CARRY

Can the result still be connected to the exact basis that shaped it?

EVIDENCE PACKET

The evidence Nexus leaves with the work

Capture enough to explain the result, inspect authority, investigate failure, and make the next decision without rebuilding the story from memory.



How to read this visual: follow an accountable agent team through evidence packet.

MECHANISM

Preserve the record needed by the next question

Agents need more than a channel through which to talk. Nexus can organise work as a pipeline when one result must feed the next. It can fan work out when several specialists should investigate in parallel. It can delegate a bounded subtask while keeping responsibility with the original owner. It can give agents a shared blackboard for structured facts and intermediate findings. It can run a bounded debate when competing interpretations need to be challenged. It can require a quorum where one opinion is not enough. These are explicit collaboration structures with defined entry conditions, handovers, merge points, and completion rules.

One confident answer is not always enough. For an important decision, Nexus can ask several specialists to examine the same evidence independently. This does not make mistakes impossible. It makes disagreement visible instead of polishing one guess until it looks certain.

- Purpose
- Basis
- Authority
- Action

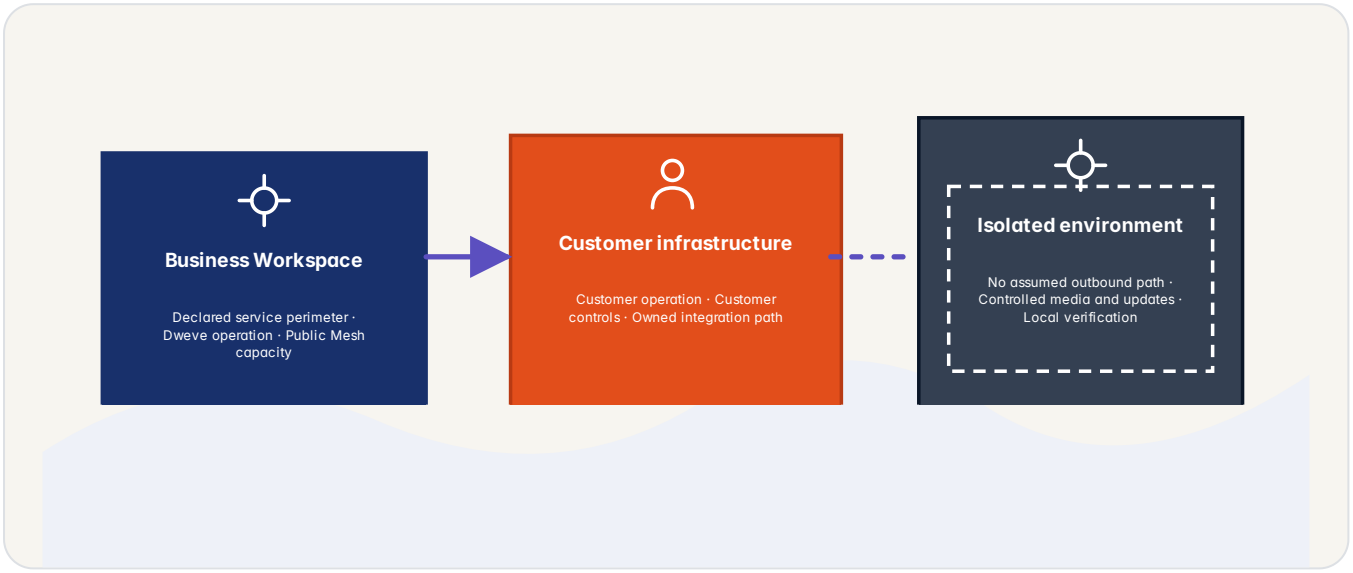
QUESTION TO CARRY

Does the retained record answer the next difficult question without reconstruction?

DEPLOYMENT CHOICE

Choose the operating posture for Nexus

Location is only one dimension. Decide who operates, who holds keys, which dependencies exist, how updates arrive, and what continuity requires.



How to read this visual: follow an accountable agent team through deployment choice.

DECISION

Compare operating responsibility, not location alone

Specialists are useful only when their work comes back together. Their internal work does not arrive as a pile of separate chat transcripts. Nexus brings the contributions back into one coherent result. Nexus can let agents work in sequence when one step depends on another. It can let them work in parallel when several parts can be investigated at the same time. It can ask several agents to examine an important question independently. It can send one difficult part to a narrower specialist. It can have a reviewer challenge an unsupported conclusion before it reaches you.

Nexus is implemented in Rust and exposed to Rust applications that want direct type-level integration, Python applications that need access from existing AI and data ecosystems, Go services that need simple deployment and operational concurrency, and Node.js applications that need Nexus inside web and event-driven systems. The bindings change how the application calls Nexus. They do not redefine agents, tasks, workflows, routing, authority, resilience, or traces.

- Begin through managed Fabric
- Run on customer infrastructure
- Close the environment
- Objective

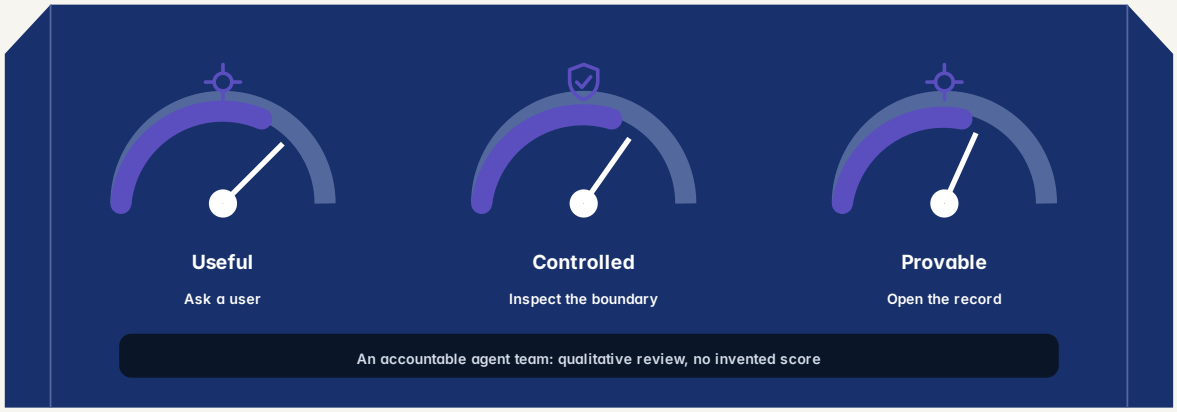
QUESTION TO CARRY

Who holds infrastructure, keys, updates, logs, recovery, and the exit path?

WHAT TO LOOK FOR

Signals that Nexus is earning trust

Progress is a useful outcome under understood control, supported by reviewable evidence.



How to read this visual: follow an accountable agent team through what to look for.

MECHANISM

Use signals that can change the next decision

Agents have health, performance, specialisation, lifecycle, and service-level state. Tasks have assignment, progress, deadline, retry, and outcome state. Workflows have active steps, blocked dependencies, branch state, gate state, and completion state. Groups have membership, leadership, composite capability, and collaboration history. The monitoring layer can record latency, success rate, pending work, failures, retries, circuit transitions, resource use, policy violations, service-level violations, delegations, lifecycle transitions, and scaling decisions.

The specialists can compare conclusions, challenge assumptions, look for missing information, and test whether the answer still holds under another interpretation. Nexus can require enough agreement before the work moves forward. When the specialists disagree, Nexus can show you what they disagree about and what evidence would resolve it.

- Useful
- Controlled
- Provable
- Objective

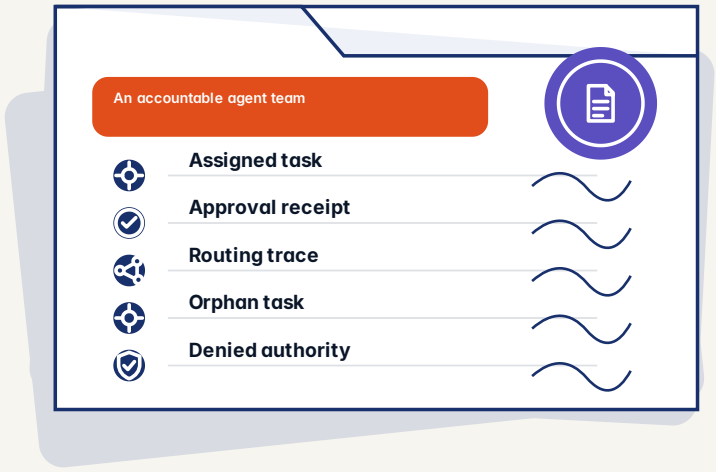
QUESTION TO CARRY

Which signals would change the next decision rather than merely impress the room?

HONEST LIMITS

Test how Nexus fails before the first success demo

Visible failure is more useful than a polished result whose limits remain unknown.



How to read this visual: follow an accountable agent team through honest limits.

DECISION

Design the failure before presenting the success

An operation should not collapse because one specialist becomes unavailable. Nexus monitors agent health, task progress, service levels, pending work, failures, resource use, and circuit state. When an agent fails, the configured response can retry the work, reroute it to another qualified agent, escalate it to a supervisor, pause the operation, or return a controlled failure.

Retry state uses bounded policies and deterministic fixed-point backoff calculations, and resource budgets ensure that repeated recovery does not become an unbounded workload of its own. The audit stream records the original failure, recovery decision, replacement participant, and eventual result. No silent fallback. No recursive agent loop without a ceiling.

- Orphan task
- Denied authority
- Failed hand-off

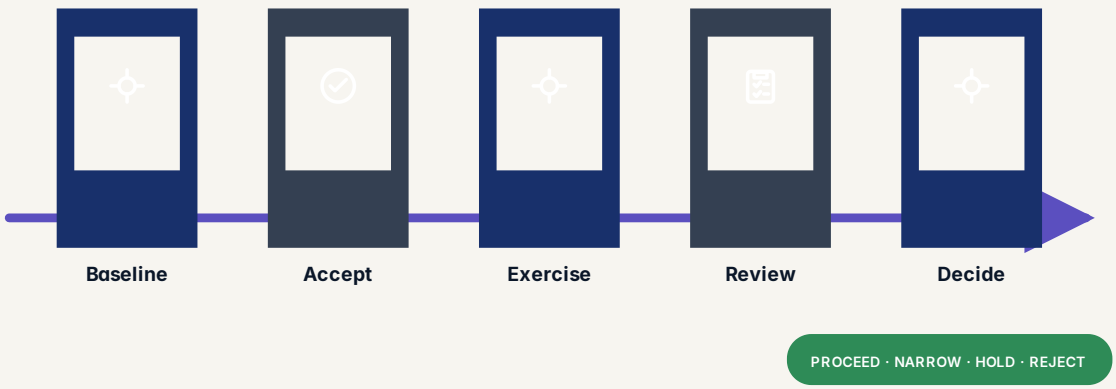
QUESTION TO CARRY

What happens when a source, permission, target, model, or supplier is unavailable?

ACCEPTANCE PATH

Close the Nexus proof with a decision

Agree the decision path before the demonstration so a strong or weak result can change what happens next.



How to read this visual: follow an accountable agent team through acceptance path.

MECHANISM

Close the proof with an owned decision

Nexus does not store what an organisation remembers as dense floating-point vectors. Every concept is a binary hypervector. Binding is XOR. Similarity is an integer bit count, the same arithmetic that staffs the organisation. There is no floating point in any hot path, so a recall result is a number you can inspect, not a distance you have to trust.

Nexus uses typed messages and events for communication between agents and runtime components. Task requests, acceptance, rejection, results, claims, releases, health events, lifecycle changes, and custom events have explicit representations. The event bus supports filtering, forwarding, dropping, and transformation, and conversation state is tracked separately from task state. The cognitive protocol layer includes A2A-compatible types for messages, tasks, artifacts, capabilities, skills, roles, and statuses.

- Baseline
- Accept
- Exercise
- Review

QUESTION TO CARRY

Are success, counter-metrics, failure cases, and stop conditions agreed first?

ADOPTION

Adopt Nexus through owned choices

Move from one bounded result to a repeatable operating capability without making the scope vague.

How to read this visual: follow an accountable agent team through adoption.

DECISION

Expand only what proved reusable

Nexus makes its own operating decisions reproducible. Capability encoding, binary routing, task ordering, workflow structure, policy evaluation, fixed-point scoring, retry calculation, resource accounting, lifecycle transitions, and local symbolic reasoning can be pinned to explicit inputs and versions.

The kernel is divided into explicit subsystems: routing matches work to capability, scheduling assigns and tracks tasks, orchestration executes workflow graphs, collaboration structures multi-agent work, communication moves typed messages and events, cognition connects perception, memory, and reasoning, governance applies policy and validates lifecycle transitions, resilience handles retry, rerouting, escalation, and controlled failure, and monitoring tracks health, service levels, and execution behaviour. Each subsystem can be tested independently without reducing the architecture to one opaque agent loop.

- Prove one workflow
- Own the operation
- Repeat the pattern
- Add responsibility

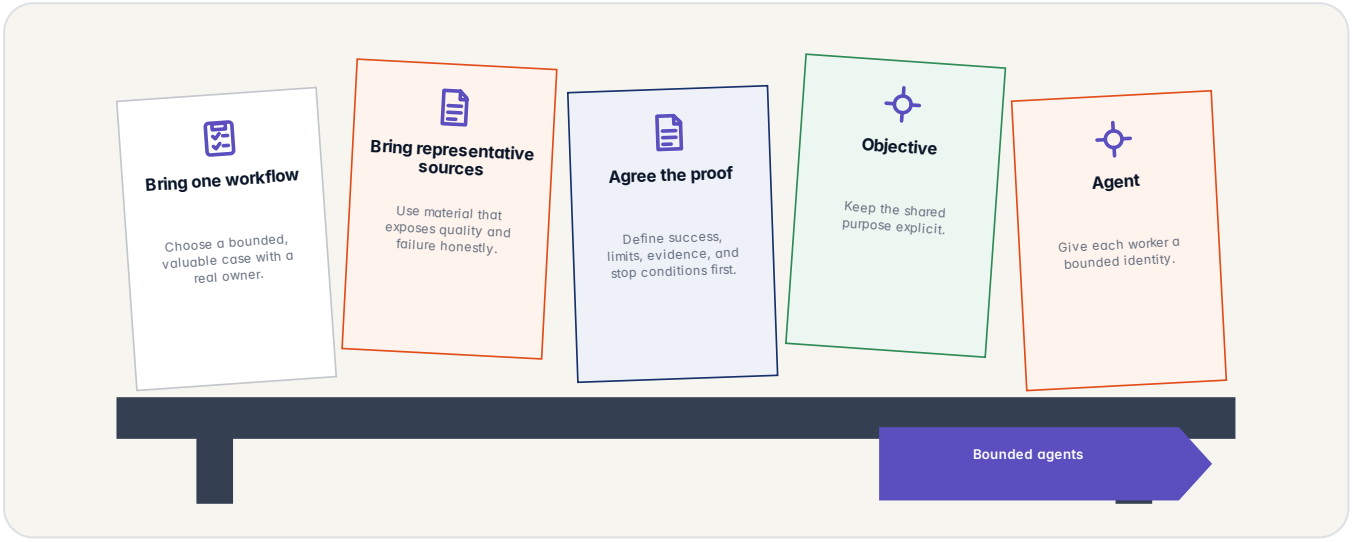
QUESTION TO CARRY

What can be reused safely, and what requires a new decision and proof?

WHAT TO PREPARE

What to bring to the Nexus working session

A small amount of real context produces a better conversation than a broad catalogue of hypothetical features.



How to read this visual: follow an accountable agent team through what to prepare.

KEEP IT BOUNDED

Sensitive production data is not required for the first conversation. Bring representative material and the rules that define a useful result.

MECHANISM

Bring enough reality to make the session useful

A simple question may need only one capable agent. A difficult document may need a reader, a policy specialist, and a writer. A research task may need several independent researchers and one reviewer. A complicated plan may need a scheduler, a budget checker, a risk specialist, and an editor. An unexpected issue may cause Nexus to bring in another specialist while the work is already underway.

Nexus keeps the task separate from the helper currently working on it. Depending on the situation, it can try again, assign the work to another qualified specialist, pause and ask what you want to do, or return a clear failure. The rest of the work does not disappear because one part had a problem. You should not need to explain everything again because one helper had a bad afternoon.

- Bring one workflow
- Bring representative sources
- Agree the proof
- Objective

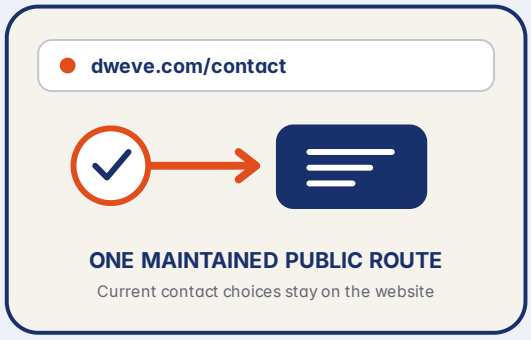
QUESTION TO CARRY

Which real sources, rules, owners, and examples will make the first session honest?

THE INVITATION

Continue the Nexus conversation

Scope a briefing, demonstration, or proof around the workflow and decision that matter to your organisation.



CONTINUE ONLINE

dweve.com/contact

Current route and contact choices

START A CONVERSATION

Continue through the current contact page

Scope a workflow or ask a technical question, then use the contact page to begin.

START	**dweve.com/contact**	Use the maintained public contact route
BRING	**One bounded question**	Workflow, audience, decision, and desired evidence
CHOOSE	**Briefing · demonstration · PoV**	Select the smallest useful next step
VERIFY	**Current scope and terms**	Confirm availability, pricing, and responsibilities before commitment

DECISION

Turn the brochure into one bounded next step

Nexus records the parts of the process that matter to you. This is not a stream of hidden model thoughts. Private chain-of-thought is not a dependable receipt. Nexus shows the observable work: sources, roles, handovers, actions, approvals, and outcome. Which files and sources were used. Which specialists contributed. Which tools were called. Which rule allowed or stopped an action. Which approval was requested. What was eventually done.

An agent may know how to send a payment, delete a record, contact a customer, modify a contract, or publish a decision. That does not mean it is authorised to do so. Nexus treats authority as part of the operating model. Policies determine which agents may perform which actions, against which resources, within which budgets, and under which conditions.

- Start: dweve.com/contact
- Bring: One bounded question
- Choose: Briefing · demonstration · PoV
- Verify: Current scope and terms

QUESTION TO CARRY

What is the smallest useful next conversation Dweve should prepare?