J

The products.

Nine products built on one cognitive runtime. Eight are at MVP and one is a proof of concept. Each addresses a problem that exists today, in a market that has not solved it, and each is built and running.

OVERVIEW

The portfolio

One runtime, nine products, nine distinct markets. The runtime is what they share; the market problem is what separates them.

J OS · J Agent	Cognitive runtime and service layer — the platform everything else is built on	MVP
J Mobile · J CLI	The persistent agent on the phone and at the command line	MVP
J Verify	Software verification: runs a project's own toolchain, never a model-generated command	MVP
J Security	Security operations: evidence correlated into contextual risk, inside your own environment	MVP
J Alert	Monitoring that delivers prioritised observations with their context, not threshold notifications	MVP
Optimaizer	AI enterprise architect: what we run, what it costs, what breaks if we change it	MVP
CacheMed	Patient-governed medical record with consent-gated access and an append-only ledger	MVP
drJ AI	Clinical reasoning over the patient-governed record; advisory to the clinician	POC
NXDOne	ERP in which the agent proposes and an authorised person commits	MVP

The ask is for bringing the MVP products to market. The architecture is built; what remains is the route to the customer.

AI capability is rented, and nothing accumulates on your side

Every enterprise AI deployment today is a subscription to somebody else's endpoint.

Capability is rented, not held	The customer subscribes to a hosted endpoint. The vendor holds the runtime, the state, the memory and the audit record. Ending the contract ends the capability and takes the accumulated knowledge with it.
Nothing compounds on the customer's side	Each invocation starts from zero. No institutional knowledge accrues anywhere the institution controls, so the second year is worth no more than the first.
There is nowhere to build	An enterprise can prompt a hosted model or wrap it. There is no runtime with attention, memory and enforceable authority to build a genuine application against.
Data leaves the estate in order to be reasoned over	For regulated sectors this is the blocking constraint, and it is not resolved by a contract clause.

1 · WHAT WE SOLVE

J OS · J Agent

MVP

A cognitive runtime the customer holds, deployed on their own infrastructure, with a service layer applications are built against.

WHAT IT IS

J OS is the runtime: attention arbitration, memory economics, planning, reflection, persistence, and the Capability Kernel that mediates every action. J Agent is the service layer that exposes it — the interface applications and interfaces call into.

HOW IT WORKS

Single-tenant deployment. The instance, the memory and the audit record sit inside the customer's estate. Applications install onto that instance under declared capability manifests; there is no central store, no gatekeeper and no shared execution surface.

WHAT CHANGES

Capability is owned rather than rented, and the knowledge accumulated by using it is an institutional asset. Data does not leave the estate to be reasoned over, which is what makes deployment possible in the sectors that need it most.

This is the platform. The eight products that follow are applications on it, and each is also a demonstration that the platform carries real work.

The assistant in your pocket is a search box with manners

Every mainstream assistant is a stateless request handler wearing an app icon.

It forgets you between sessions	Context ends when the conversation does. What you explained last week has to be explained again, and the tenth exchange is no better informed than the first.
Each interface holds its own version of you	Phone, desktop and browser each maintain separate context. There is no single entity to talk to, only several instances with divergent views.
It works only while you are looking at it	Nothing runs when the app is closed. It cannot pursue anything, cannot come back with a finding, and cannot do work between the times you open it.
It cannot act on anything that matters	Either it has no authority at all, or it has an unmediated tool list nobody would grant access to real systems.

2 · WHAT WE SOLVE

J Mobile · J CLI

MVP

The same persistent agent, reached from the phone and from the terminal — one memory, one identity, continuing to work while the app is closed.

WHAT IT IS

J Mobile is the flagship interface and the route to market: J in the pocket, with the full runtime behind it. J CLI is the same entity at the command line, for developers and for operations.

HOW IT WORKS

Interfaces are deliberately stateless — transport to the agent, not a second locus of state. The cognitive cycle runs on the runtime whether or not an interface is open, so investigations continue and results are waiting when you next look.

WHAT CHANGES

One entity that knows you, across every device, that carries yesterday into today and returns with things you did not ask for. That is a different product category from an assistant, and it is the one consumers will recognise first.

J Mobile is the commercial entry point: the lowest-friction way for a customer to encounter the architecture before buying the platform.

AI writes the code, and the same system certifies it works

Verification in AI-assisted development is currently an assertion, not an execution.

Correctness is asserted by the author	The model that generated the code reports that it is correct. No independent execution stands behind the claim, and the failure mode is a confident statement rather than an error.
Model-generated commands run against real repositories	Assistants construct shell invocations as text and execute them. The blast radius is the developer's machine or the CI runner, and the command was never reviewed by anyone.
Test and documentation debt is invisible until release	Coverage, drift between code and documentation, and untested paths surface at the point they are most expensive.
Nothing produced is admissible as evidence	A transcript of an assistant saying the tests passed is not an artifact anyone can hand to an auditor or a client.

3 · WHAT WE SOLVE

J Verify

MVP

Verification by execution: J Verify detects the stack and runs the project's own toolchain, and never a command a model wrote.

WHAT IT IS

A J OS application for validation, testing and documentation. It identifies the stack, runs the project's existing test and analysis tooling, and produces a record of what was executed and what it returned.

HOW IT WORKS

Commands are owned by per-stack adapters, never generated by the model. Execution is capability-authorised per project through the kernel, untrusted code runs containerised, and every run is recorded with its inputs and its result.

WHAT CHANGES

The output is evidence rather than an assertion: what ran, on what revision, and what it returned. That is defensible to a client, an auditor or a regulator, and it is the property the whole category currently lacks.

The architecture's authority separation is the product feature here — the reasoning layer proposes what to verify; it cannot decide what gets executed.

Security signals arrive from every direction and mean nothing separately

Logs, vulnerabilities, authentication events, configuration changes, telemetry and incident evidence arrive through different systems.

The signals are fragmented across tools	Each source is accurate and partial. Correlating them is manual work performed under time pressure by the people who can least afford to spend it, and the correlation is never written down.
An investigation ends when the query does	Security tooling answers one question at a time and retains nothing between them. What was established an hour ago has to be established again, so investigations are as shallow as the shift allows.
Alerts are counted, not assessed	Severity is assigned by rule and decoupled from business context, so the finding that matters sits in the same queue, at the same priority, as the noise generated around it.
Analysis means surrendering the environment	Managed detection requires telemetry, and often access, to leave the organisation. In regulated sectors that requirement is the reason the answer is no.

J Security

MVP

Security intelligence that keeps investigations open, correlates evidence into probable impact, and runs inside your environment rather than requiring you to hand it over.

WHAT IT IS

A security operations application on J OS. Vulnerabilities, events, configuration, identity activity and operational evidence become contextual risk analysis across four functions: threat analysis, incident intelligence, risk and posture, and audit and compliance.

HOW IT WORKS

Investigations persist across invocations and accumulate evidence rather than resetting per query. Reasoning is separated from execution authority: J Security runs as observer, analyst, investigator or authorised operator, according to what the kernel has been granted — not according to configuration.

WHAT CHANGES

Probable impact instead of an alert count. Incident timelines that assemble themselves, vulnerabilities tied to business context, and a decision record that is audit evidence at the moment it is created rather than reconstructed afterwards.

The authority model is the commercial argument: the same product runs read-only inside a regulated estate and as an authorised operator elsewhere, and the difference is enforced rather than promised.

An alert tells you a number moved, and nothing else

Monitoring notifies on thresholds. Interpretation is left entirely to whoever happens to be on call.

The alert is the end of the information, not the start	A threshold was crossed. Whether it matters, what changed recently, and whether it relates to anything else is work the operator reconstructs from memory, at speed, usually at night.
Signals are siloed by layer	Application health, infrastructure, dependencies, certificates, backups and scheduled tasks are watched by separate tools with separate notions of severity, and nothing reads them together.
Volume produces fatigue, then suppression	Undifferentiated alerts get filtered, muted and eventually ignored. The outage is very often preceded by an alert somebody had already switched off for good reason.
Nothing connects an alert to a deployment or a finding	The same latency spike means one thing after a release and another after a configuration change. Threshold monitoring cannot distinguish them, because it holds no context to distinguish them with.

J Alert

MVP

Monitoring that produces a prioritised observation with its context attached, rather than a threshold notification with none.

WHAT IT IS

A modular monitoring layer for J OS. It watches applications — health, exceptions, jobs, releases, runtime failures; infrastructure — CPU, memory, disks, containers, Kubernetes, cloud services; and dependencies — databases, queues, APIs, DNS, certificates, backups, scheduled tasks.

HOW IT WORKS

Detected changes enter J's cognitive environment as observations rather than notifications: correlated against recent deployments, configuration changes, prior findings and open investigations, then prioritised by the same attention arbitration as anything else competing to be looked at. Runs standalone or alongside J Security.

WHAT CHANGES

The same latency spike is read differently after a release than after a configuration change, and arrives already interpreted. An alert that carries its context is an alert that stays switched on, which is the only property that ultimately matters.

J Alert is where the architecture is most directly visible: novelty detection decides what is worth raising, attention economics decides what is raised first.

No organisation can state what it runs, what it costs, or what breaks

Estate knowledge lives in people's heads, stale registers, and a consultant's slide deck from two years ago.

Nobody can state the estate	The configuration database was accurate on the day it was populated. Every answer about what is running is assembled by asking people, and the people who knew have left.
Cost is attributed to invoices, not to systems	Spend is known per vendor and per contract, not per application or per capability. The question of what a given system actually costs to run has no owner and no answer.
Change impact is discovered in production	Nothing models the dependencies, so the consequences of a migration, a decommission or a version change are found by attempting it.
The analysis leaves with the consultants	Enterprise architecture is bought as a deliverable. It is a snapshot, it is out of date within a quarter, and the model that produced it was never handed over.

6 · WHAT WE SOLVE

Optimaizer

MVP

An AI enterprise architect that maintains the model continuously, holds it inside the institution, and answers change-impact questions before the change.

WHAT IT IS

A J OS application that builds and maintains a live model of the estate: what runs, what depends on what, what it costs, and what the consequences of a change would be.

HOW IT WORKS

Evidence is gathered from actual sources rather than from interviews. Anomalies and contradictions in the estate become investigations the agent opens on its own, and the model is revised as evidence changes rather than rebuilt from scratch.

WHAT CHANGES

The architecture function becomes continuous rather than periodic, and the model is an asset the institution owns. Change-impact questions are answered before the migration, not discovered during it.

Attention economics and endogenous investigation are load-bearing here: the estate is far larger than any budget to examine it, so what gets looked at has to be chosen well.

Your medical record is fragmented, and you do not control access to it

The record exists in pieces, in institutions, under governance the patient does not hold.

The record is fragmented across every provider who treated you	Each institution holds a partial view. No one holds the whole, least of all the patient, and reconciliation happens by fax, by portal, or not at all.
Consent is a signature on admission, not a control	A form signed once authorises broad access indefinitely. There is no mechanism by which a patient grants, scopes or withdraws access to a specific party for a specific purpose.
Access logs exist, and belong to the institution being audited	The record of who looked is held and maintained by the same party whose access is in question, in a system the patient cannot see.
Emergency access is unbounded or absent	Break-glass is either not implemented, or implemented as unlimited access with a note in a log nobody reads.

7 · WHAT WE SOLVE

CacheMed

MVP

One record, governed by the patient: access granted by them, scoped, revocable, and provable afterwards on a ledger nobody can edit.

WHAT IT IS

A single patient-governed medical record. Providers ingest into it and read from it; the patient holds the consent controls; every access is gated and recorded.

HOW IT WORKS

A consent gate mediates every read. The audit ledger is append-only and enforced at the database, not by application code. Break-glass access is bounded, attributed and flagged. Clinical data is ingested through standard health-data interchange.

WHAT CHANGES

The patient holds the whole record and the authority over it, and every party who looked can be shown, provably, after the fact. That is a governance property no portal currently provides.

CacheMed deliberately contains no AI runtime. The separation between the record and anything that reasons over it is the design decision, not an omission.

Clinical decision support is either ignored or inadmissible

The clinician carries the liability, and neither available option supports them in carrying it.

Rules engines produce alert fatigue	Population-level rules fire on individual patients, most alerts are irrelevant, and the documented clinical response is to switch them off.
A model's conclusion cannot be interrogated	A generated recommendation carries no derivation. Asked why, the system generates a new plausible explanation rather than retrieving the reasoning that produced the original.
Neither option is admissible where liability sits	The clinician is accountable for the decision. Support that cannot be examined, challenged, or shown to have used the right evidence cannot be relied on in that position.
The tooling has no access to the whole record	Reasoning over a partial view produces confident conclusions from missing data, which is worse than no conclusion at all.

drJ AI

POC

Reasoning over the patient-governed record that proposes with its evidence attached, and leaves the decision with the clinician by construction.

WHAT IT IS

A J OS application performing clinical reasoning over the CacheMed record. It surfaces findings, proposes, and shows what it used and why. It does not decide, and it cannot act.

HOW IT WORKS

Every access to the record passes the same consent gate as any other party, and is recorded on the same ledger. Conclusions are stored as beliefs with evidence, confidence and derivation, so a proposal can be interrogated later exactly as it stood.

WHAT CHANGES

Support that can be examined and challenged, in a form that fits where the liability actually sits. The clinician can ask why, and receives the original reasoning rather than a fresh account of it.

At proof of concept. The clinical validation pathway, not the engineering, is what stands between this and deployment.

The ERP records decisions it played no part in making

The system of record sits downstream of the work, and nobody will authorise an agent upstream of it.

The work happens outside the system of record	Analysis, reconciliation and decision-making happen in spreadsheets and inboxes. The ERP receives the result and records it, having contributed nothing to reaching it.
Automation is rigid, and the exceptions are the work	Workflow engines handle the path that was anticipated. Everything else falls out to a person, and everything else is where the effort actually goes.
Nobody will authorise an agent to touch financial records	An AI with write access to the ledger is not a governance question anyone is willing to answer, so AI in the ERP means a chatbot on top of it.
Approval is a rubber stamp or a bottleneck	Either the approver cannot see what they are approving, or the approval queue becomes the constraint on the whole process.

NXDOne

MVP

The agent proposes with its evidence; an authorised person commits. Authority stays with the human, and the proposal arrives already justified.

WHAT IT IS

An ERP with a J connector, organised around a capability-to-approval-mode taxonomy: every class of action is mapped to who may authorise it and under what conditions.

HOW IT WORKS

The agent performs the analysis and produces a proposal carrying the evidence and derivation behind it. The kernel mediates: the agent cannot commit. An authorised person reviews a justified proposal and commits it, and the commit is recorded against them.

WHAT CHANGES

The reasoning moves inside the system of record while the authority stays with the person accountable for it. Approval becomes a real decision on a justified proposal rather than a rubber stamp on an opaque one.

This is the clearest commercial demonstration of authority separation: it is the property that makes an agent admissible in a finance function at all.

INVESTMENT

The ask: bringing the MVP products to market

The engineering risk is behind us. Eight products are at MVP, built and running on a runtime that works, and the remaining distance is commercial rather than architectural. Funding is sought to cross it.

Production hardening and certification	Scale, load and failure behaviour under real conditions, and the compliance and security work each regulated market requires before a first deployment — the difference between a product that runs and one an institution may buy.
First reference deployments	Paying deployments with named institutions in health, finance and enterprise IT. Each MVP needs the customer that proves it in production, and that is what converts an architecture into a market position.
Commercial capacity around the platform	Sales, onboarding, support and documentation. Nine products currently share one architect; nothing about that scales, and it is the binding constraint on revenue rather than on capability.
drJ AI through clinical validation	The single proof of concept in the portfolio. What it needs is the clinical validation pathway and a partner institution, not further engineering.

Built. Running. Not yet in the market.

Nine products across health, finance, enterprise IT, security operations and software engineering, each solving a problem that exists today and is not solved by anything on the market. Eight are at MVP and one is a proof of concept. What they need now is the route to the customer.

Simona D. Thrussell, PhD · NXD Tech · Tilburg

sdthrussell@gmail.com · +31 6 15 37 00 63 · jecosystem.com