MICKAI™

THE FIFTY BRAINS · A SOVEREIGN INTELLIGENCE OPERATING SYSTEM

Contents

Introduction: what the Science and Engineering subsystem is

Mickai is the British Sovereign Intelligence Operating System, a SIOS. It runs frontier-class artificial intelligence entirely on hardware the operator controls, under keys the operator holds, with a complete and cryptographically verifiable record of everything the system does. It is held privately by its founder, Micky Irons. The substrate primitives are filed at the UK Intellectual Property Office under the GB2607309.8 to GB2611702.8 patent family, named inventor Micky Irons. This ebook is about one part of that operating system: the Science and Engineering subsystem, and the five specialist brains inside it.

A Sovereign Intelligence Operating System is organised the way an operating system is organised, into subsystems, and each subsystem contains specialist brains scoped to a body of work. The Mickai cooperative runs domain brains across five subsystems: Intelligence and Defence, Science and Engineering, Health and Humanity, Culture and Heritage, and Knowledge and Exploration. Beneath those five sits a sixth layer, the Chronus orchestration kernel, which holds the cognitive mechanics that move work between specialists: routing, planning, tool use, retrieval, long-term memory, the audit ledger, identity, quorum, permissions, and revocation. A deterministic conductor routes each fragment of an operator's request to the brain that owns it, sequences the resulting calls in a fixed order so the audit chain can be replayed, and signs every decision at the moment of commit. The brains do not freelance. They are scoped, identified, signed, and audited.

The word brain is used precisely here, and it is worth pausing on, because it is the first thing that distinguishes the Mickai architecture from the systems it is most often compared to. A brain in the Mickai sense is a domain specialist with its own scoped knowledge base, its own cloned tooling, its own signed identity on the internal bus, and its own declared responsibilities. It is not a prompt, not a persona, and not a routing weight. Where a mixture-of-experts model gates a single set of parameters through a softmax and produces one undifferentiated stream, the Mickai cooperative dispatches a request to a named, isolated specialist whose every action is attributable to it and to it alone. For the science and engineering buyer that attributability is not an abstraction. It is the difference between a calculation you can put your name to and a number a chatbot produced that nobody can trace, and in a domain where a number ends up in a structural design, a grid setpoint, a peer-reviewed derivation, or a board report, the question after the fact is never merely

what did the system compute. It is which component computed it, against what source, by what method, and can that be shown.

The Science and Engineering subsystem is the one a working engineer, a research group, an infrastructure owner, or an analytics function reaches for when the task is technical and the output has to survive scrutiny. There are five brains in it, each named for a figure of force, lightning, scale, or rigour, and each scoped to a distinct slice of the technical world:

- **JAXON**, the computer-science specialist. It reads repositories, writes patches, refactors across many files, synthesises tests, and runs code in a sandbox, all on-device so the operator's source never leaves the machine.
- **RAIDEN**, the real-time-systems specialist. Electrical and grid engineering, weather modelling, alerting, and emergency-response coordination, built around hard deadlines and signed timestamps.
- **QUANTUM**, the hard-sciences specialist. Physics, mathematics, and proof-carrying derivations, where the chain of reasoning is itself a verifiable artefact a peer reviewer can replay.
- **TITAN**, the engineering and infrastructure specialist. Structural analysis, capital-project planning, materials selection, and safety-margin calculation, with every design change recorded as a deterministic action that has a declared inverse.
- **KARP**, the data and analytics specialist. Spreadsheets, dataframes, query results, and board-ready reporting, with signed transformations and access control that descends to the individual cell.

Why a UK engineering, research, or industrial buyer cares

Start with the constraint a UK engineering or research buyer actually lives under, because it is the constraint that makes a sovereign substrate matter rather than a sovereign substrate sound appealing. The work is technical, the inputs are often confidential, and the output has a long life. A structural calculation done today may be challenged in fifteen years. A grid-protection setting put in place this morning may be examined after an incident this evening. A derivation submitted to a journal will be checked by a referee who is paid to disbelieve it. A board report drawn from an analytics pipeline will be relied on for a capital decision and then audited. In every one of those cases the technical answer is necessary but not sufficient. The

buyer also needs to be able to show, later, how the answer was reached, from what source, by what method, and that it has not been altered since.

For most AI categories that requirement is an inconvenience. For science and engineering it is the whole game. An engineer cannot put a number into a design and tell the regulator that a cloud model produced it and the prompt is gone. A research group cannot submit a derivation and tell the referee that the working was generated by a vendor's chatbot and cannot be reproduced. An analytics function cannot hand the board a figure and be unable to say which query against which source produced it. The technical professions have always run on traceability, on the idea that a claim is only as good as the working behind it, and the working has to be available. AI that cannot produce the working is, for this domain, AI that cannot be used on the work that matters.

There is a second constraint that sits underneath the first, and it is sharper still for the industrial buyer. The inputs are frequently things that must not leave the operator's perimeter. Source code is the clearest case: a software engineer at a defence prime, a fintech, or any firm with proprietary algorithms cannot send a repository to a third-party server to be refactored, because the repository is the firm's value and the server is outside the firm's control. The same is true of grid-topology data at a distribution network operator, of unpublished experimental results at a research institute, of materials specifications at a manufacturer, and of the underlying rows in a financial dataset. The cloud is not merely a place where these things might leak. It is a place where, once sent, they are subject to a legal and jurisdictional regime the operator does not control. The sovereign answer is not to encrypt the data in transit to someone else's machine. It is to do the work on the operator's own machine and never let the data leave at all.

The Science and Engineering subsystem is built for exactly that buyer, the one who needs frontier technical capability and cannot give up either provenance or perimeter to get it. Three properties run through all five brains and are worth stating once at the front, because the chapters return to them repeatedly.

First, **everything is signed**. Every patch, every grid alert, every derivation, every design recommendation, and every analytical report is signed at the moment of commit under FIPS 204 ML-DSA-65, the United States NIST post-quantum digital signature standard finalised in 2024. The signature is post-quantum-secure today, ahead of the NCSC migration deadlines, and it is produced under a key the operator holds in hardware, not a key a vendor holds in a cloud. An engineer can hand a regulator a signed calculation. A researcher can attach a signed derivation to a

submission. The signature does not say the answer is correct. It says this exact output came from this exact operator's system at this exact time and has not changed since, which is the foundation correctness has to be argued on top of.

Second, **everything is traceable**. The signed records append to a hash-linked chain, the Open Audit Record (OAR), under SHA-3-512 hash-linking. Each new entry carries the hash of the one before it, so the chain cannot be reordered, back-dated, or quietly edited without the break showing. A reviewer can walk back from a board figure, a structural result, or a published derivation to the inputs and the method it was built from, and can do so in a browser-resident verifier with only a public key, offline, with no recourse to the vendor. This is what Mickai means by trust-domain externalisation: the audit chain lives under the operator's key in an open format, so the operator, the regulator, the peer reviewer, and any third party can replay the same chain at once and reach the same verdict independently. The verifier does not phone home. It does not need the AI vendor to be in business. It needs the public key and the chain, and it produces a deterministic verdict.

Third, **everything is reversible where the work is consequential**. The science and engineering domain is full of actions that change state: a multi-file code refactor, a structural redesign, a grid reconfiguration, a data transformation. Two of these brains carry the compensating-inverse discipline filed in the patent corpus, so that a change is recorded together with the action that would undo it, and a paper change can be retraced and rolled back even after downstream work has begun. The pre-commit simulation primitive lets a risky multi-file change be reviewed as a diff against the target state before it commits at all. The subsystem assumes that engineering is iterative and that mistakes are part of the method, and it makes the record of every change, and the path back from it, a first-class property rather than an afterthought.

The rest of this ebook takes the five brains one at a time. Each chapter opens with the brain's image, then explains what the brain does, walks its declared responsibilities and the sources and tools it works with, situates it inside the subsystem and against the regulatory and standards frame, works through two or three operator scenarios in the verticals where the brain earns its place, and closes with a short note on the meaning of its name and a handful of frequently asked questions. The final chapter draws the five together, describes the audit substrate that sits beneath all of them, and offers a procurement note for the buyer who has to put this in front of a technical-assurance committee.

A word on what this ebook does not do. It does not invent capabilities. Every responsibility, every knowledge source, and every tool named below is drawn from the canonical Mickai brain catalogue. Where a patent is referenced, it is referenced because the brain's own entry references it, and it is described in terms of what it contains, never in terms of who did or did not help file it. Patents in the Mickai corpus are filed at the UK Intellectual Property Office, not granted, and they are spoken of here as filings. The science and engineering domain is a poor place to overstate a system's reach, because the people reading the output are trained to check it, so the account here is deliberately held to what the substrate actually carries.

Chapter One: JAXON, the computer-science brain

Source that never leaves the machine

JAXON is the computer-science specialist of the Mickai cooperative, and its domain in the catalogue is stated cleanly: computer science, software, algorithms. Its one-line description is the entire promise in a sentence. It reads, writes, refactors, and tests code on-device, and the source never leaves the machine. That last clause is the one that matters, and it is worth holding onto through the whole chapter, because it is the property that separates JAXON from every cloud coding assistant a software engineer has been offered, and it is the property a regulated or proprietary-code shop cannot do without.

The work JAXON does is the work a strong software engineer does. It reads repositories. It generates patches. It refactors across many files at once. It synthesises tests. It executes code in a sandbox and reviews the resulting diffs. It designs algorithms and reasons about their complexity. None of this is novel as a list of capabilities, and JAXON does not pretend otherwise. What is novel is the substrate it does the work on. The repository is read on the operator's own hardware. The patch is generated on the operator's own hardware. The tests run on the operator's own hardware. At no point is a line of the operator's source transmitted to a vendor's server to be processed, and at no point does the operator have to trust a vendor's assurance that the code was not retained, logged, or used to train a model.

The assurance is structural rather than contractual. The code cannot be misused off the premises because it never goes off the premises.

What JAXON is responsible for

The Mickai catalogue gives JAXON four declared responsibilities, and read together they describe a complete software-engineering loop rather than an autocomplete.

Repository reading, file editing, and multi-file refactors. JAXON does not work a single file at a time the way a line-completion tool does. It reads across a repository, builds an understanding of how the pieces relate, and makes coordinated edits that span many files. A refactor that renames a concept used in forty places, splits a module that has grown too large, or threads a new parameter through a call graph is a multi-file change by nature, and JAXON treats it as one operation rather than forty disconnected ones. This is the difference between a tool that helps you type and a specialist that helps you change a system.

Test generation and sandbox execution. JAXON synthesises tests for the code it reads and writes, and it runs them in a sandbox. The sandbox matters for two reasons. The first is safety: code that is being generated and changed should be executed in an isolated environment before it touches anything real. The second is evidence: the sandbox run produces a result, and the result is recorded. A patch that JAXON proposes does not arrive as an unverified suggestion. It arrives with the tests it was checked against and the outcome of running them, so the engineer reviewing it is reviewing a change that has already been exercised rather than a guess.

Algorithm design and complexity analysis. Beyond reading and editing existing code, JAXON does the genuinely computer-scientific work of designing algorithms and reasoning about their complexity. This is where the brain's knowledge base earns its place. An engineer can ask JAXON not merely to write a function but to choose the right approach, to weigh the time and space trade-offs, and to explain why one structure is preferable to another for the case at hand. The output is not only the code but the reasoning that justifies it, and the reasoning is grounded in the canonical computer-science literature the brain treats as authoritative.

Pre-commit dry-run for risky changes. This is the responsibility that ties JAXON most directly to the Mickai substrate, and it rests on the pre-commit simulation primitive filed as patent 15. A risky multi-file change can be reviewed as a diff against the target state before it commits. The engineer sees exactly what the

repository will look like after the change, file by file, line by line, and decides whether to let it proceed, all before a single byte is written to the working tree. The dry-run is a safety rail for precisely the kind of sweeping change that is most useful and most dangerous, the change that touches many files at once and would be tedious to unpick by hand if it went wrong.

What JAXON reads, and what it works in

JAXON's competence comes from its sources, and the catalogue names them. Its knowledge base is the canon of computer science and software engineering: the ACM Digital Library, the arXiv computer-science preprint archive, Cormen, Leiserson, Rivest and Stein's *Introduction to Algorithms*, Knuth's *The Art of Computer Programming*, the IEEE Computer Society publications, the Stack Overflow question-and-answer corpus, an index of public GitHub repositories, the RFC archive, the POSIX specification, and the ECMAScript and Python language specifications. The shape of that list tells you what kind of engineer JAXON is. It is grounded in the foundational texts and the formal specifications, the sources a serious practitioner would cite, rather than in folklore. When JAXON reasons about an algorithm's complexity it is reasoning from CLRS and Knuth. When it reasons about a language feature it is reasoning from the language specification. When it reasons about a protocol it is reasoning from the relevant RFC. The authority is citable, and that is what lets JAXON's outputs be defended rather than merely accepted.

JAXON works through a set of software-engineering tools cloned into the brain. It uses Codex, the sovereign plain-text graph personal-knowledge-management surface that every Mickai brain shares, as its knowledge spine. On top of that it runs Compass, a code-intelligence surface that gives it structured understanding of a codebase rather than a flat text view; Mast, an integrated-development-environment surface where the editing actually happens; Quill, a programmable plain-text editor; Marble, an architecture-diagram canvas for reasoning about and communicating system structure; and Loom, a build-and-test directed-acyclic-graph scheduler that sequences the compile and test steps in dependency order. The tools matter because they are how JAXON's discipline is enforced rather than merely intended. Compass means JAXON understands the code structurally before it changes it. Loom means the build and test steps run in the right order and their results are captured. Marble means a large refactor can be reasoned about at the level of architecture rather than only at the level of text.

Where JAXON sits in the subsystem

JAXON does not work alone. Its catalogue entry places it tightly against the Chronus orchestration layer, because the git, build, and test invocations that software work depends on are tool-use operations, and tool use is a Chronus mechanic. When JAXON commits a patch, runs a build, or executes a test suite, those actions are dispatched through the Chronus tool-use orchestration and signed into the audit chain like any other action. Within the Science and Engineering subsystem JAXON is the brain the others lean on for implementation. QUANTUM produces a numerical method and hands the implementation to JAXON. TITAN needs a piece of analysis automated and JAXON writes it. KARP needs a transformation expressed as code rather than a one-off query and JAXON expresses it. JAXON is the subsystem's hands for anything that has to become running software.

How every action is signed into the OAR

This is the section the regulated software buyer reads most carefully, so it is worth being concrete. Every consequential action JAXON takes is signed at the moment of commit under FIPS 204 ML-DSA-65 and appended to the Open Audit Record, the hash-linked chain held under the operator's key. When JAXON generates a patch, the patch is signed. When it runs the test suite, the test invocation and its result are signed. When it executes code in the sandbox, the execution and its outcome are signed. When a pre-commit dry-run is reviewed and accepted, the acceptance is signed. Each of these entries carries the SHA-3-512 hash of the entry before it, so the sequence is tamper-evident: a record cannot be inserted, removed, reordered, or back-dated without breaking the hash chain, and the break is visible to anyone who replays it.

The consequence is a software-engineering audit trail of a kind that ordinary version control does not provide. A git history tells you what changed and who the commit is attributed to. The OAR tells you that this exact change was produced by this operator's JAXON at this exact time, was checked against these exact tests with these exact results, and has not been altered since, and it tells you so in a form a third party can verify offline with only the public key. For a shop that has to demonstrate to an assurance function or a client that its AI-assisted code was produced under control and has a provable lineage, this is the difference between an assertion and a proof. The verifier is browser-resident and offline. It does not need the Mickai vendor to exist. It needs the chain and the key, and it returns a deterministic verdict.

Operator scenarios

A defence-prime software team under export control. A software group at a defence prime maintains a codebase that is subject to export control and cannot be shared outside a closed network, let alone uploaded to a commercial coding assistant. The group still wants the throughput that AI-assisted development offers. With JAXON the repository is read and refactored entirely on the prime's own hardware inside the closed network, and the source never leaves it. When a developer asks for a multi-file refactor, JAXON proposes it as a pre-commit dry-run, the developer reviews the diff against the target state, and only on acceptance does the change commit, with the whole sequence signed into the OAR. When the prime's software-assurance function later asks how a particular change was made and on what authority, the answer is a signed chain that walks from the request to the dry-run to the accepted commit to the test results, verifiable offline. The group gets frontier coding assistance without ever creating the export-control exposure that a cloud assistant would create, and without ever losing the provenance the assurance function requires.

A fintech protecting proprietary algorithms. A trading or risk firm's edge is in its algorithms, and those algorithms are in its source code. Sending that code to a third-party model to be improved is structurally unacceptable, because the code is the value and the third party is outside the firm's control. JAXON lets the firm's quantitative developers refactor, test, and extend that code on the firm's own machines with the source held inside the perimeter. The complexity analysis JAXON provides, grounded in CLRS and Knuth, helps the developers reason about the performance characteristics of a strategy's implementation without consulting anyone outside. Every change is signed, so if a regulator later asks the firm to demonstrate the controls around its AI-assisted development, the firm has a tamper-evident record rather than a policy document. The proprietary algorithm stays proprietary, and the development of it stays provable.

A research-software group reproducing a result. Reproducibility is a chronic problem in computational research, where a published result may depend on code that has drifted, on an environment that has changed, or on a step nobody recorded. A research-software group using JAXON gets reproducibility as a property of the substrate. When JAXON generates and runs the analysis code, the code, the environment, the test invocations, and the results are signed into the OAR in sequence. A collaborator, a journal, or a funder who needs to confirm that a result was produced by a particular version of the code against particular inputs can replay

the chain and see exactly that, offline, with only the public key. The group does not have to reconstruct after the fact what it did, because the substrate recorded it as it happened, in a form a third party can verify.

Regulatory and standards relevance

JAXON's relevance to the standards frame is twofold. The first is the substrate's own cryptographic standing: the signing pipeline uses FIPS 204 ML-DSA-65, a finalised NIST post-quantum standard, and the hash-linking uses SHA-3-512, so the audit chain is built on published, inspectable cryptographic standards rather than a proprietary scheme. For a buyer whose technical-assurance process asks what algorithms underpin a control, those are concrete and checkable answers. The second is the brand of evidence JAXON produces. Software-development standards and secure-development lifecycle frameworks increasingly ask organisations to demonstrate control over how code is produced and to maintain auditable records of change. JAXON's signed, hash-linked record of every patch, test, and execution is exactly the kind of evidence those frameworks ask for, produced automatically as a by-product of the work rather than assembled afterward by hand. The brain does not claim to certify a codebase against any particular standard. It produces the provable record that any such certification has to be built on.

What this brain does not do

JAXON is a computer-science specialist, and it is disciplined about its edges. It does not push the operator's source code to any external service, and that is the point of it, so any reading of JAXON that imagines cloud processing has misunderstood the brain entirely. It does not reach into the physical-engineering domain that belongs to TITAN, the real-time signal domain that belongs to RAIDEN, or the hard-sciences derivation work that belongs to QUANTUM, beyond implementing what those brains hand it. It does not silently apply sweeping changes: the pre-commit dry-run exists precisely so that a risky multi-file change is reviewed before it commits rather than after. And it does not produce code whose provenance is unrecorded. Every consequential action is signed into the OAR, which means JAXON cannot make an untraceable change even if asked to, because tracing is not an optional mode, it is how the brain operates.

Frequently asked questions

Does JAXON ever send our source code to a server? No. JAXON runs entirely on-device, on hardware the operator controls. Reading, refactoring, test generation, and sandbox execution all happen locally, and the operator's source never leaves the machine. This is the defining property of the brain, and it is structural rather than a configuration option, which is why JAXON is usable in environments where a cloud coding assistant is disqualified outright.

What is the pre-commit dry-run, and why does it matter? The pre-commit dry-run, built on patent 15, lets a risky multi-file change be reviewed as a diff against the target state before it commits. The engineer sees exactly what the repository will look like after the change, file by file, and decides whether to proceed before anything is written. It is the safety rail for large coordinated changes, the kind that are most useful and most painful to unpick by hand if they go wrong.

How is JAXON different from a cloud coding assistant? Two ways that matter. First, the source stays on the operator's machine, so there is no external exposure of proprietary or controlled code. Second, every consequential action is signed into the Open Audit Record under a post-quantum signature and hash-linked, so the operator has a tamper-evident, offline-verifiable record of how every change was produced. A cloud assistant offers neither the perimeter nor the provable lineage.

Can a third party verify how a change was made without trusting us or the vendor? Yes. The OAR is held under the operator's key in an open format, and the verifier is browser-resident and offline. A third party with the public key can replay the chain from the request through the dry-run to the accepted commit and the test results, and reach a deterministic verdict, with no recourse to the operator's good word or to the vendor's servers.

Chapter Two: RAIDEN, the real-time-systems brain

The brain built around a deadline

RAIDEN is the real-time-systems specialist of the Mickai cooperative, and its domain is the set of problems where the answer is only useful if it arrives in time. The catalogue scopes it to real-time systems, electrical engineering, weather, and emergency response, and its one-line description names the four together: real-time signals, electrical and grid engineering, weather, and emergency response. Every other brain in the subsystem can tolerate latency. A code refactor can take a minute. A derivation can take an hour. RAIDEN cannot, because the things RAIDEN reasons about, a grid fault, a developing storm, an emergency unfolding, do not wait for a leisurely computation. RAIDEN is the brain built around hard deadlines and signed timestamps, and that single difference in design philosophy shapes everything about it.

The contrast the catalogue draws is exact and worth repeating, because it is the heart of the chapter. Where most brains can tolerate seconds of latency, RAIDEN is built around hard deadlines and signed timestamps. A hard deadline is a deadline that, if missed, makes the result wrong rather than merely late. An alert that a transformer is about to fail is worthless if it arrives after the transformer has failed. A storm warning is worthless if it arrives after the storm. RAIDEN treats time as a first-class constraint, schedules its work against deadlines, and stamps its outputs with signed

timestamps so that downstream actors can see not only what RAIDEN concluded but exactly when it concluded it. In the real-time domain, when is part of the answer.

What RAIDEN is responsible for

The Mickai catalogue gives RAIDEN four declared responsibilities, and they trace the path of a real-time signal from the sensor to the response.

Real-time signal ingestion with deadline scheduling. RAIDEN ingests signals in real time, and it schedules the work against deadlines rather than processing it whenever it gets around to it. This is the property that makes RAIDEN a real-time system rather than a fast one. A real-time system is not merely quick; it is predictable under load, because it schedules work so that the most time-critical computation is guaranteed its slot. RAIDEN's deadline scheduling means that when many signals arrive at once, the one that has to be acted on first is acted on first, by design, rather than by luck. The POSIX real-time extensions and the RTOS specifications in its knowledge base are the foundations of exactly this discipline.

Electrical-grid and weather-model integration. RAIDEN integrates with the two real-time domains the catalogue names most prominently: the electrical grid and the weather. On the grid side it works with the data and the standards of power-system engineering, distributed energy resources, substation automation, and reliability. On the weather side it works with operational meteorological feeds and reanalysis data. These are not separate competences accidentally placed in one brain. They are unified by their shared character: both are real-time physical systems, both generate continuous signal streams, both have states that change faster than a human can manually track, and both have consequences when a developing condition is missed.

Signed emergency alerts with provenance. When RAIDEN raises an alert, the alert is signed, and the signature carries provenance. This is where the substrate meets the emergency-response mission, and it rests on the hardware-bound key filed as patent 08. Every alert RAIDEN emits is signed with a key bound to the hardware, so that a downstream actor, an operator, a control room, a responding agency, can verify that the alert genuinely came from a Mickai system and was not spoofed. In an emergency the cost of acting on a false alert and the cost of ignoring a true one are both high, and a spoofed alert is a recognised attack against emergency and infrastructure systems. RAIDEN's signed provenance is the answer: an alert that cannot be forged because forging it would require the hardware-bound key, which the attacker does not have.

Coordination handoffs to SALVATOR and TITAN. RAIDEN does not act on the physical world by itself. When its signals indicate a humanitarian incident it hands off to SALVATOR, the humanitarian brain in the Health and Humanity subsystem, and when they indicate an infrastructure response it hands off to TITAN, the engineering and infrastructure brain in its own subsystem. The handoffs are themselves recorded and signed, so the path from a real-time signal through RAIDEN's analysis to the brain that acts on it is a continuous, auditable chain. RAIDEN is the sensor and the alarm. SALVATOR and TITAN are part of the response.

What RAIDEN reads, and what it works in

RAIDEN's authority comes from the standards and feeds of the real-time physical domains, and the catalogue names them. Its knowledge base spans IEEE 1547 for distributed-energy-resource interconnection, IEC 61850 for substation automation, the NERC reliability standards, the Met Office weather-data feeds, the NOAA climate data, the ECMWF reanalysis, the ICS-CERT advisories for industrial-control-system security, the ITU recommendations, the POSIX real-time extensions, and the VxWorks and RTOS specifications. The list reads like the reference shelf of a power-systems and real-time-control engineer, because that is what RAIDEN is. The grid standards, IEEE 1547, IEC 61850, NERC, are the standards a distribution network operator and a substation engineer work to. The weather sources, Met Office, NOAA, ECMWF, are the operational and reanalysis feeds a meteorological or emergency function relies on. The ICS-CERT advisories ground RAIDEN in the security posture of the industrial-control systems it touches. And the real-time-operating-system specifications, POSIX real-time and the RTOS canon, are the foundation of its deadline discipline.

RAIDEN works through a focused set of real-time tools cloned into the brain. It uses Codex, the shared sovereign plain-text graph knowledge surface, as its spine. On top of that it runs Pulse, a real-time signal pipeline, which is the instrument through which the live signal streams flow and are scheduled against deadlines; Prism, an operational-dashboards surface, which is where the state of the monitored systems is made visible to an operator; and Watchtower, an alert-and-emergency-feed aggregator, which gathers the inbound alerting picture. The toolset is deliberately lean, and the leanness is itself a design statement. A real-time brain does not want a sprawl of heavyweight surfaces between the signal and the response. It wants a tight pipeline, a clear dashboard, and a reliable alert aggregator, and that is what RAIDEN carries.

Where RAIDEN sits in the subsystem

RAIDEN's place in the cooperative is defined by its handoffs. Its catalogue entry names two: its output stream feeds SALVATOR on humanitarian incidents and TITAN on infrastructure response. SALVATOR sits in the Health and Humanity subsystem, which tells you something important about the architecture, namely that the subsystem boundaries are organisational rather than walls. A real-time signal that becomes a humanitarian matter crosses from Science and Engineering into Health and Humanity through a signed handoff, and the audit chain follows it across. TITAN sits in RAIDEN's own subsystem and is the brain that turns a real-time infrastructure signal into an engineering response, a structural assessment, a capital-project implication, a materials question. RAIDEN is the brain that watches the live physical world, and it is wired to the brains that act when the world demands action.

How every action is signed into the OAR

RAIDEN's signing has a property the other brains' signing does not emphasise as strongly: the timestamp is load-bearing. Every alert and every consequential action RAIDEN takes is signed under FIPS 204 ML-DSA-65 and appended to the Open Audit Record, hash-linked under SHA-3-512, exactly as the rest of the subsystem. But because RAIDEN works to deadlines, the signed timestamp on each record is not merely metadata. It is part of what the record proves. After an incident, the question is frequently not only what did the system detect but when did it detect it, and how long was it between the detection and the response. RAIDEN's signed, hash-linked, time-stamped chain answers that question with a record that cannot be back-dated, because back-dating an entry would break the hash link to the entry that genuinely came after it. The alert's hardware-bound signature, from patent 08, proves the alert was genuine, and the chain's ordering proves the timeline was as recorded.

For an infrastructure operator this is precisely the evidence an incident investigation needs. When a grid event, a flood, or an emergency is examined afterward, the operator can produce a signed, tamper-evident timeline showing exactly when each signal was received, when RAIDEN raised each alert, and when each handoff to a responding brain occurred, all verifiable offline with only the public key. The investigation does not have to reconstruct the timeline from logs of uncertain integrity. It can replay a chain whose ordering is cryptographically fixed.

Operator scenarios

A distribution network operator watching the grid. A distribution network operator runs a network whose state changes continuously as load, generation, and faults come and go, and increasingly as distributed energy resources, rooftop solar, batteries, electric-vehicle charging, push and pull power in ways the network was not originally designed for. RAIDEN ingests the real-time signals from the substations, grounded in IEC 61850 and IEEE 1547, schedules the analysis against deadlines so the most time-critical condition is evaluated first, and surfaces the state on a Prism dashboard. When a developing fault crosses a threshold, RAIDEN raises a signed alert with provenance, and the control-room operator can verify the alert genuinely came from the network's own Mickai system rather than a spoofed source, which matters because spoofed control-system messages are a known attack. If the event escalates to an infrastructure response, RAIDEN hands off to TITAN, and the whole sequence, signal, alert, handoff, is signed into the OAR with timestamps that cannot be back-dated. After the event, the operator hands the regulator a tamper-evident timeline rather than a reconstruction.

An emergency-response coordination cell during a storm. A flood or storm response is a real-time problem with high stakes and many actors, and one of its chronic difficulties is trusting the information flowing in. RAIDEN integrates the operational weather feeds, Met Office, ECMWF reanalysis for context, with the inbound alert picture aggregated through Watchtower, and produces signed alerts as the situation develops. The signature matters because in a fast-moving emergency a false alert can divert resources and a spoofed one can do real harm, and RAIDEN's hardware-bound signing means the responding agencies can verify each alert is genuine. Where a signal indicates a humanitarian incident, RAIDEN hands off to SALVATOR, crossing the subsystem boundary through a signed handoff. The cell ends the response with a signed, time-ordered record of what was known when, which is exactly what the post-incident review will ask for and exactly what is hardest to assemble after the fact.

A critical-infrastructure operator defending an industrial-control network. Industrial-control systems are real-time by nature and increasingly under threat, and the ICS-CERT advisories in RAIDEN's knowledge base ground it in that threat picture. An operator running RAIDEN against a control network gets real-time monitoring whose alerts are signed and provenance-bearing, so that the alarm itself cannot be spoofed, an important property when a sophisticated adversary's goal may be precisely to inject false telemetry or suppress true alarms. RAIDEN's deadline scheduling means the monitoring stays predictable under load, including under the load an attacker might deliberately generate. Every alert and handoff is

signed into the OAR, giving the operator a defensible record for the security and safety regulators who oversee critical national infrastructure. The brain does not replace the control system's own safety functions. It adds a sovereign, signed layer of real-time analysis and alerting on top of them.

Regulatory and standards relevance

RAIDEN sits directly on top of the standards that govern the real-time physical systems it watches, and that is its first relevance: it reasons in the vocabulary of IEEE 1547, IEC 61850, and the NERC reliability standards on the grid side, and of the operational meteorological feeds on the weather side, so its outputs are expressed in terms the relevant engineers and regulators already use. Its second relevance is to the security and resilience regimes that govern critical national infrastructure. Frameworks in this space, and the regulators that enforce them, increasingly require operators to demonstrate not only that they detected an incident but that their detection and alerting were trustworthy and their timeline is reconstructable. RAIDEN's signed, hardware-bound alerts and its tamper-evident, time-ordered audit chain are built for exactly that requirement. The ICS-CERT grounding means RAIDEN treats the control systems it touches as part of a contested security surface rather than a benign one. As with the rest of the subsystem, RAIDEN does not certify compliance with any particular regime. It produces the signed, timed evidence that compliance and incident investigation are argued on.

What this brain does not do

RAIDEN is a real-time signal and alerting brain, and it is careful about its boundaries. It does not itself carry out the physical-engineering response to an infrastructure event, that is TITAN's work, nor the humanitarian response to an incident, that is SALVATOR's; RAIDEN detects, analyses, alerts, and hands off. It does not perform the offline, latency-tolerant computation that belongs to QUANTUM or the code work that belongs to JAXON, because its design is organised around deadlines rather than depth. It does not emit unsigned alerts, because an unsigned alert in this domain is a liability, and the hardware-bound signature is the mechanism that makes RAIDEN's alerts trustworthy. And it does not replace the dedicated safety-instrumented and protection systems that an infrastructure operator runs; it is a sovereign analytical and alerting layer that sits alongside them and adds signed provenance and an auditable timeline, not a substitute for engineered safety functions.

Frequently asked questions

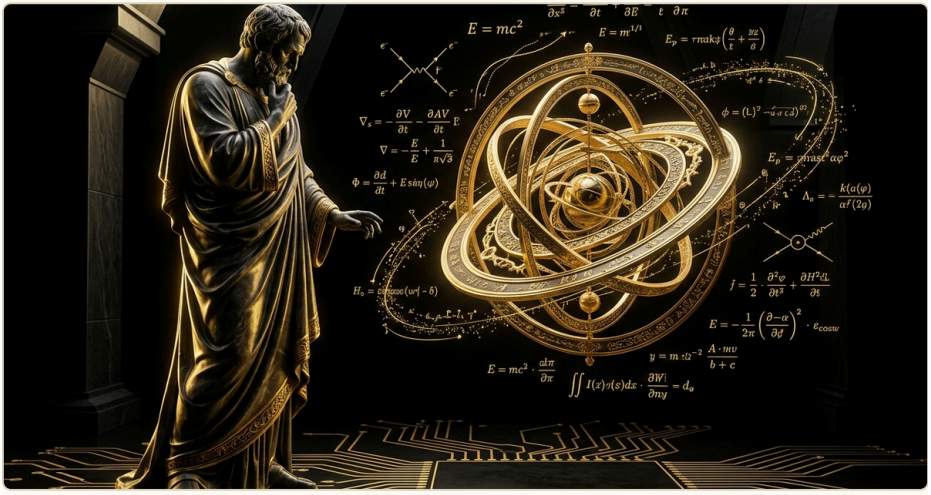
What does it mean that RAIDEN is built around hard deadlines? It means RAIDEN schedules its work so that the most time-critical computation is guaranteed to run in time, rather than processing signals whenever it happens to get to them. A hard deadline is one that makes the result wrong if missed, like an alert that arrives after the event it warned of. The POSIX real-time extensions and RTOS foundations in RAIDEN's knowledge base are the basis of this deadline scheduling, which is what makes RAIDEN a real-time system rather than merely a fast one.

Why are RAIDEN's alerts signed, and what does the signature prove? Each alert is signed with a hardware-bound key, from patent 08, so a downstream actor can verify the alert genuinely came from the operator's Mickai system and was not spoofed. Spoofed alerts are a real attack against emergency and infrastructure systems, where both acting on a false alert and ignoring a true one are costly. The signature makes the alert unforgeable without the hardware-bound key, which an attacker does not have.

How does RAIDEN help after an incident, not just during one? Every alert, analysis, and handoff RAIDEN performs is signed and appended to the hash-linked OAR with a timestamp that cannot be back-dated, because back-dating would break the chain. After an incident the operator can replay a tamper-evident, time-ordered record showing exactly what was detected when and when each response was triggered, verifiable offline with only the public key. That is precisely the timeline a post-incident review needs and the hardest thing to assemble from ordinary logs.

Does RAIDEN replace our grid-protection or control-system safety functions? No. RAIDEN is a sovereign, signed layer of real-time analysis, monitoring, and alerting that sits alongside an operator's engineered safety and protection systems. It adds provenance and an auditable timeline. It does not substitute for the dedicated protection relays, safety-instrumented systems, or control logic an infrastructure operator runs, and it hands off the physical response to TITAN and the humanitarian response to SALVATOR rather than carrying them out itself.

Chapter Three: QUANTUM, the hard-sciences brain



Where the working is the artefact

QUANTUM is the hard-sciences specialist of the Mickai cooperative, scoped in the catalogue to physics, mathematics, and the hard sciences, with a one-line description that names its most distinctive property: physics, mathematics, and the hard sciences, with symbolic, numerical, and proof-carrying outputs. The phrase to hold onto is proof-carrying. Most AI systems produce an answer and, if pressed, a plausible-sounding explanation. QUANTUM produces an answer whose chain of reasoning is itself a verifiable artefact, a thing a peer reviewer can take apart and replay rather than a narrative they have to take on faith. In the hard sciences the working is not a courtesy attached to the result. The working is the result, because a physical or mathematical claim with no reproducible derivation is not a finding, it is an assertion.

The work QUANTUM does spans the two registers the hard sciences run on: the symbolic and the numerical. Symbolic work is the manipulation of expressions, the derivation of a result in closed form, the algebra and calculus and proof that produce an equation rather than a number. Numerical work is computation, the evaluation of a model, the propagation of values and their uncertainties through a calculation to produce a quantity with an error bar. QUANTUM does both, and crucially it binds both to the audit ledger so that a peer reviewer can replay the computation step by step. That replay capability is the chapter's central claim, and it is worth being

precise about what it means. It does not mean QUANTUM shows its working in prose. It means the steps are recorded as signed entries in an ordered chain, so that the derivation can be re-executed and checked against the record, by someone who was not present and does not trust the operator's word.

What QUANTUM is responsible for

The Mickai catalogue gives QUANTUM four declared responsibilities, and they describe the full arc of hard-sciences work from computation to peer review.

Symbolic and numerical computation. QUANTUM works in both registers, the symbolic and the numerical, and the pairing is deliberate. A great deal of physics and mathematics lives in the movement between the two: a result derived symbolically is evaluated numerically, a numerical observation suggests a symbolic relationship, and a serious piece of work usually requires fluency in both. QUANTUM carries that fluency, with the symbolic-computation surface and the numerical tooling cloned into the brain, and it treats the two not as separate modes but as the two halves of a single competence.

Proof-carrying derivations and uncertainty propagation. This is QUANTUM's signature responsibility. A derivation is proof-carrying when the chain of reasoning is recorded as a verifiable artefact rather than asserted as a conclusion. Uncertainty propagation is the discipline of carrying error through a calculation so that the final quantity arrives with an honest error bar rather than a spurious precision. The two go together because both are about not overstating what is known. A proof-carrying derivation does not ask the reader to trust that the steps are valid; it lays them out so they can be checked. Uncertainty propagation does not present a number as exact when it is not; it carries the uncertainty so the number's reliability is visible. Together they are the hard-sciences definition of intellectual honesty, mechanised.

Domain handoff to applied engineering and astrophysics. QUANTUM does not pretend to do everything the physical sciences touch. When a result needs to be applied in engineering it hands off to TITAN, and when the domain is cosmology or astrophysics it coordinates with EXFINITUM, the relevant brain in the Knowledge and Exploration subsystem. The handoffs keep QUANTUM scoped to what it owns, the underlying physics and mathematics, while connecting it to the brains that take the physics into application or into the specialised domains of the very large.

Signed replay records for peer-review reproduction. The output of QUANTUM's work is not only an answer; it is a signed replay record, a chain a peer reviewer can use to reproduce the computation step by step. This is the responsibility that turns proof-carrying from an aspiration into a mechanism. Because QUANTUM's outputs are signed under the audit ledger and ordered in a hash-linked chain, a reviewer can take the record and walk the derivation, confirming each step against the one before it. Peer review in its ideal form is reproduction, and reproduction requires that the working be available in a form that survives transmission to a sceptic. QUANTUM's signed replay record is that form.

What QUANTUM reads, and what it works in

QUANTUM's authority comes from the canonical literature of the hard sciences, and the catalogue names it. Its knowledge base spans the arXiv mathematics and physics preprint archives, INSPIRE-HEP for high-energy physics, the *Physical Review* series, the American Mathematical Society's MathSciNet, the NIST CODATA fundamental constants, the IUPAC nomenclature, the Particle Data Group Review, Mathematical Reviews and Zentralblatt MATH, the Springer Nature physics archives, and, distinctively, the Stanford Encyclopedia of Philosophy for the philosophy of science. The shape of that list is exactly the reference base of a working physicist or mathematician, with two details worth drawing out. The CODATA fundamental constants and the IUPAC nomenclature mean QUANTUM works from the internationally agreed values and naming, not from approximations, which matters when a calculation has to be reconciled with everyone else's. And the inclusion of the philosophy of science is not decoration. The hard sciences rest on methodological commitments, about what counts as evidence, what a model is for, and when a result is established, and a brain that produces proof-carrying work is a brain that has to take the philosophy of method seriously.

QUANTUM works through a set of scientific tools cloned into the brain. It uses Codex, the shared sovereign plain-text graph knowledge surface, as its spine. On top of that it runs Stele, a physics-and-maths citation graph, which binds claims to the literature so a derivation's references resolve to real sources; Slate, a symbolic-computation surface, which is where the algebraic and analytic work happens; Quill, here in its role as a mathematical-typesetting and notebook surface, which is where derivations are written up in the notation the field actually uses; and Aperture, an instrument-and-observation primitive, which connects QUANTUM to the data side of physics, the measurements and observations a theory is tested against. The toolset mirrors how a hard-sciences researcher actually works: the literature graph for

citation, the symbolic surface for derivation, the notebook for write-up, and the instrument primitive for the link to observation.

Where QUANTUM sits in the subsystem

QUANTUM's relationships are stated precisely in its catalogue entry. It coordinates with JAXON on numerical implementation, with TITAN on engineering application, and with EXFINITUM on cosmology and astrophysics. The first of these is the most frequent in practice: a numerical method that QUANTUM specifies often has to become running code, and JAXON is the brain that writes it, on-device and signed. The second connects pure physics to the built world: when a physical result has an engineering consequence, TITAN carries it into structural or mechanical application. The third reaches into the Knowledge and Exploration subsystem, where EXFINITUM owns the domains of cosmology and astrophysics, the physics of the very large and the very distant. QUANTUM is the subsystem's source of underlying physical and mathematical truth, and it is wired to the brains that implement it, apply it, and extend it into the specialised reaches of the universe.

How every action is signed into the OAR

QUANTUM's signing is the mechanism behind its proof-carrying promise, so it deserves a careful statement. Every derivation step, every numerical computation, and every replay record QUANTUM produces is signed under FIPS 204 ML-DSA-65 and appended to the Open Audit Record, hash-linked under SHA-3-512. The hash linking is what makes the derivation reproducible in the strong sense: because each step carries the hash of the step before it, the chain cannot be reordered or have a step quietly inserted or removed, so the derivation a reviewer replays is provably the derivation QUANTUM actually performed. The signature binds the chain to the operator's key, so the reviewer knows whose system produced it. And the replay record is the artefact the reviewer works from, a signed, ordered sequence that re-executes to the same result.

The consequence for the hard sciences is a kind of reproducibility that ordinary computational science struggles to achieve. The reproducibility crisis in computational research is, at bottom, a crisis of lost working: the code drifted, the environment changed, the parameters were not recorded, and the result can no longer be regenerated. QUANTUM's signed replay record attacks that problem at the root by recording the working as it happens, in a tamper-evident chain, so that the derivation does not have to be reconstructed later because it was never lost. A

referee, a collaborator, or a funder can replay the chain offline with only the public key and confirm that the result follows from the stated steps. The proof is not a claim about the working. The proof is the working, signed.

Operator scenarios

A research physics group preparing a submission. A research group is preparing a result for a journal, and the result rests on a derivation that combines symbolic work with a numerical computation carrying propagated uncertainties. With QUANTUM the derivation is performed on the group's own hardware, with each step signed into the OAR and the references resolved through Stele to the literature. When the group submits, it can attach a signed replay record, so that the referee, whose job is to disbelieve until convinced, can reproduce the derivation step by step rather than taking the working on trust. If a numerical method has to become reusable code, QUANTUM hands the implementation to JAXON, and that code is signed too. The group gets a submission whose working is reproducible by construction, which is the strongest position a piece of computational science can be in, and it gets it without the unpublished result ever leaving the group's perimeter.

A national-laboratory team reconciling against agreed constants. A team at a national laboratory is doing precision work where the answer has to be reconciled with internationally agreed values, and small inconsistencies in constants or nomenclature can derail a comparison. QUANTUM works from the NIST CODATA fundamental constants and the IUPAC nomenclature in its knowledge base, so its calculations are expressed in the same agreed values everyone else uses, and its uncertainty propagation carries the errors honestly through to the final quantity. Every step is signed, so when the result is shared with collaborators at other institutions, they receive not a number to be trusted but a replay record to be reproduced. The team's confidence in a cross-institutional comparison rests on the fact that the working is verifiable rather than on the reputation of whoever produced it.

A mathematics group recording a proof. A mathematics group has produced a proof and needs it to be checkable, communicable, and durable. QUANTUM performs the symbolic work in Slate and writes it up in Quill in the field's own notation, binding the cited results through Stele to MathSciNet and the literature. The proof is recorded as a signed, ordered chain in the OAR, so a colleague who was not present can replay the argument step by step and confirm each follows from the last, with the hash linking guaranteeing the steps are in the order they were actually

performed. Years later, when the proof is cited or challenged, the group can produce the signed replay record and demonstrate, offline and with only the public key, that the argument stands as recorded. The proof's durability is a property of the substrate rather than of anyone's memory or filing.

Regulatory and standards relevance

QUANTUM's relevance to the formal frame is of a slightly different character from the other brains, because the hard sciences are governed less by regulators and more by the standards of reproducibility, citation, and agreed values that the scientific community enforces on itself. QUANTUM is built for exactly those standards. Its grounding in the CODATA constants and IUPAC nomenclature means it works from the internationally agreed reference values, so its results are reconcilable with the global literature rather than expressed in a private idiom. Its proof-carrying outputs and signed replay records are a direct answer to the reproducibility expectations that funders, journals, and research-integrity frameworks increasingly impose, expectations that a computational result be regenerable and its working be available. And because the signing pipeline rests on FIPS 204 ML-DSA-65 and SHA-3-512, the integrity of the replay record is itself built on published cryptographic standards, so the claim that a derivation has not been altered is a checkable cryptographic claim rather than an assurance. QUANTUM does not adjudicate scientific correctness, that is what peer review is for. It makes the working reproducible so that peer review can do its job.

What this brain does not do

QUANTUM is a hard-sciences specialist, and its discipline shows in what it declines. It does not carry physics into engineering application on its own, that handoff goes to TITAN, nor into cosmology and astrophysics, which belong to EXFINITUM; QUANTUM owns the underlying physics and mathematics and connects to the brains that take it further. It does not write the production code for a numerical method, it specifies the method and hands the implementation to JAXON. It does not assert results without a replayable derivation, because an assertion without working is precisely what proof-carrying output is designed to refuse. And it does not present numbers as more certain than they are, because uncertainty propagation is one of its declared responsibilities, and a spurious precision is, in the hard sciences, a kind of error. QUANTUM's restraint is the point: a brain that produces proof-carrying work cannot also be a brain that takes shortcuts, because the shortcut would be visible in the proof.

Frequently asked questions

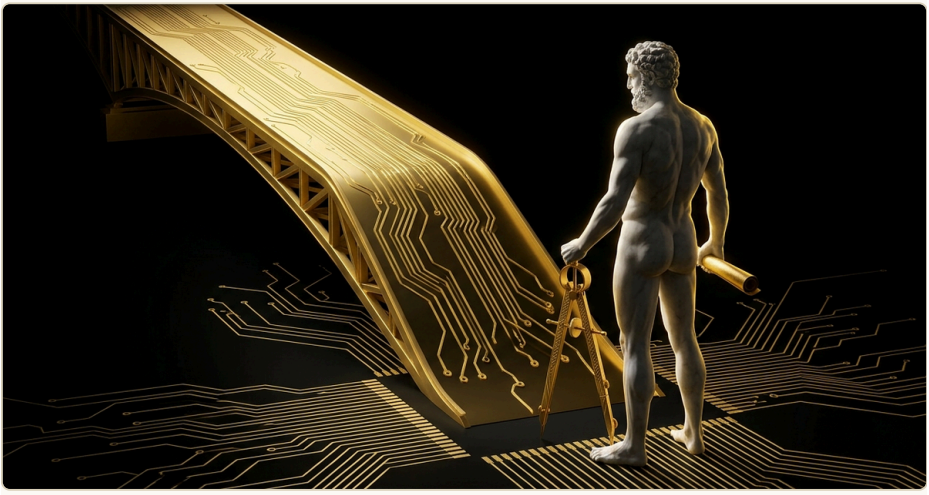
What does proof-carrying actually mean here? It means QUANTUM's reasoning is recorded as a verifiable artefact rather than asserted as a conclusion. The derivation steps are signed and ordered in a hash-linked chain, so a peer reviewer can replay the computation step by step and confirm each step follows from the last. The proof is not a prose explanation of the working; it is the working itself, in a form a sceptic can reproduce offline with only the public key.

How does QUANTUM help with reproducibility? The reproducibility crisis in computational science is largely a crisis of lost working. QUANTUM records the working as it happens, signed and hash-linked, so the derivation is never lost and does not have to be reconstructed later. A referee, collaborator, or funder can replay the signed record and regenerate the result, which turns reproducibility from an aspiration that depends on careful record-keeping into a property of the substrate.

Does QUANTUM decide whether a scientific result is correct? No. QUANTUM produces symbolic and numerical work with proof-carrying derivations and propagated uncertainties, and it makes the working reproducible. Whether a result is correct is what peer review establishes, and QUANTUM's contribution is to make that review possible by ensuring the working can be replayed and checked. It adjudicates nothing; it makes adjudication possible.

Why does QUANTUM's knowledge base include the philosophy of science? Because the hard sciences rest on methodological commitments about evidence, models, and what counts as an established result, and a brain that produces proof-carrying work has to take method seriously. The Stanford Encyclopedia of Philosophy grounding is not decoration; it reflects that QUANTUM's discipline of not overstating what is known, of carrying uncertainty honestly and laying out working for inspection, is itself a methodological stance the brain is built to hold.

Chapter Four: TITAN, the engineering and infrastructure brain



Every change has a way back

TITAN is the engineering and infrastructure specialist of the Mickai cooperative, scoped in the catalogue to engineering, construction, and infrastructure, with a one-line description that names its range: an engineering and infrastructure specialist, from structural analysis to capital-project planning. TITAN is the brain that takes the physical world as its subject, the world of structures, materials, civil and mechanical design, and the large projects that build and maintain the built environment. If QUANTUM owns the underlying physics, TITAN owns the application of it to things that have to stand up, carry load, and last.

The work TITAN does is the work of an engineer responsible for things that fail consequentially. It performs structural analysis. It plans capital projects and resolves their dependencies. It does civil and mechanical design. It selects materials. It calculates safety margins. And it does all of this under a discipline the catalogue states precisely and that gives this chapter its title: TITAN treats every design recommendation as a deterministic action with a declared inverse, so that a paper change can be retraced even after physical work has begun. That phrase, a declared inverse, is the heart of TITAN, and it rests on the compensating-inverse primitive filed as patent 14. Engineering is iterative, designs change, and a change made on paper can have consequences that ripple into work already under way in the physical world. TITAN records every change together with the action that would undo it, so

that the path back is always available, even once concrete has been poured or steel has been cut.

What TITAN is responsible for

The Mickai catalogue gives TITAN four declared responsibilities, and they span the arc of an engineering project from analysis to delivery.

Structural and mechanical analysis. TITAN performs the core analytical work of structural and mechanical engineering: determining how a structure or a component behaves under load, where the stresses concentrate, whether a design will hold. This is the foundational competence on which everything else rests, because a capital plan, a materials choice, and a safety margin all depend on a correct analysis of how the thing behaves. TITAN's grounding in the Eurocode suite, the AISC steel manual, and ASCE 7 means the analysis is performed against the standards the profession actually designs to, not against a generic approximation.

Capital-project planning and dependency resolution. Beyond the analysis of a single structure, TITAN plans capital projects and resolves their dependencies. A capital project is a web of interdependent activities, each constrained by the others, and the planning of it is itself an engineering problem, one of sequencing, constraint, and resource. TITAN's grounding in the NEC4 contract suite and the ICE manual of civil-engineering construction means it plans in the framework the UK infrastructure sector actually contracts and builds within. Dependency resolution is the discipline of working out what must come before what, so that the plan is feasible rather than merely a wish-list of activities.

Materials selection with safety-margin verification. TITAN selects materials, and it verifies safety margins as part of the selection. Materials selection is never only about a material's headline properties; it is about whether the material, in the application, with the loads and the environment it will face, carries an adequate margin against failure. TITAN's grounding in the ASTM materials standards and the Engineering Toolbox reference data means it selects from real, specified materials with known properties, and its safety-margin verification means the selection is checked against the margin the design requires rather than assumed to be adequate.

Compensating-inverse change tracking. This is TITAN's distinctive responsibility, and it is the one that ties the brain most directly to the substrate. Every design recommendation is treated as a deterministic action with a declared inverse, the compensating action that would undo it, filed as patent 14. The

consequence is that a change can be retraced and reversed even after physical work has begun. In engineering this is a profound property, because the cost of a change rises steeply once construction starts, and the ability to retrace exactly what was changed, when, and how to undo it on paper is the difference between a controlled correction and an expensive investigation. TITAN makes the path back from every change a recorded, deterministic thing rather than a reconstruction.

What TITAN reads, and what it works in

TITAN's authority comes from the standards of the engineering and construction profession, and the catalogue names them. Its knowledge base spans the BS and EN engineering standards, the Eurocode structural-design suite, the AISC steel-construction manual, ASCE 7 for minimum design loads, the HSE Construction Design and Management Regulations, the ICE manual of civil-engineering construction, the ASTM materials standards, the UK Highway Design Manual, the NEC4 New Engineering Contract suite, and the Engineering Toolbox reference data. The list is the working reference shelf of a UK chartered engineer, and the inclusions are pointed. The Eurocode suite, the BS and EN standards, and the AISC manual are the structural-design canon. ASCE 7 and the Engineering Toolbox supply the loads and the reference data. The HSE Construction Design and Management Regulations ground TITAN in the legal duties around construction safety, which is not optional in UK practice but a statutory obligation. And the NEC4 suite and the ICE manual mean TITAN plans and contracts in the actual framework UK infrastructure projects use, so its capital plans speak the language a UK project manager and quantity surveyor already work in.

TITAN works through a set of engineering tools cloned into the brain. It uses Codex, the shared sovereign plain-text graph knowledge surface, as its spine. On top of that it runs Plumb, an engineering computer-aided-design surface, where the geometric design work happens; Slate, here in its role as a finite-element and computational-engineering surface, where the structural and mechanical analysis is computed; Marble, a design-canvas surface, for working through and communicating a design's structure; and Vellum, a specification workspace, where the precise, contractual specification of a design is written. The toolset mirrors the workflow of an engineering office: the CAD surface for the geometry, the finite-element surface for the analysis, the design canvas for the thinking, and the specification workspace for the document that contractors will actually build from. Notably, TITAN shares Slate with QUANTUM, which is exactly right, because the finite-element analysis TITAN runs is the applied face of the physics QUANTUM owns.

Where TITAN sits in the subsystem

TITAN is one of the most connected brains in the subsystem, because engineering touches physics, real-time monitoring, and regulation all at once. Its catalogue entry names three relationships. It integrates with QUANTUM on the underlying physics, because a structural analysis is applied physics and the two brains share the computational-engineering surface. It integrates with RAIDEN on real-time monitoring, because a structure or an infrastructure asset, once built, generates real-time signals about its condition, and RAIDEN is the brain that watches them. And it integrates with ZEUS, the legal and governance brain in the Intelligence and Defence subsystem, on regulatory compliance, because engineering in the UK is a regulated activity with statutory duties, and a design recommendation often has a compliance dimension that ZEUS owns. TITAN is the brain that designs and plans the physical world, and it is wired to the brain that supplies its physics, the brain that monitors what it builds, and the brain that weighs the law it must satisfy.

How every action is signed into the OAR

TITAN's signing has a particular force because of what it signs and because of the compensating-inverse discipline. Every structural analysis, every capital plan, every materials selection, and every design recommendation TITAN produces is signed under FIPS 204 ML-DSA-65 and appended to the Open Audit Record, hash-linked under SHA-3-512. But TITAN signs something the other brains do not: the declared inverse of each change. Because every design recommendation is recorded together with the action that would undo it, the OAR holds not merely a history of what was decided but a reversible history, a chain in which each change carries its own undo. The hash linking guarantees the order, so the sequence of changes to a design is provably the sequence that actually occurred, and a change cannot be back-dated or quietly removed without the break showing.

For an engineering organisation this produces a design record of unusual strength. When a structure is examined years later, after an incident, during a dispute, or in the ordinary course of asset management, the owner can replay a signed, tamper-evident chain showing exactly what was designed, what was changed, when, by what reasoning, and how each change could have been undone. This is precisely the record an engineering dispute or a safety investigation turns on, and it is precisely the record that is hardest to assemble after the fact from drawings of uncertain version and emails of uncertain completeness. The verifier is offline and needs only the public key. The engineering judgement is not replaced by the record, but the record

makes the judgement defensible, because it shows exactly what was decided and on what basis.

Operator scenarios

A civil-engineering consultancy on a major infrastructure project. A consultancy is designing a major piece of infrastructure under the NEC4 contract framework, with a capital plan of many interdependent activities and a design that will change repeatedly as the project develops. With TITAN the structural analysis is performed against the Eurocode suite and ASCE 7, the capital plan is built and its dependencies resolved in the NEC4 framework the project actually contracts in, and every design change is recorded with its declared inverse and signed into the OAR. When the design changes after groundwork has started, the team can retrace exactly what changed and how it would be undone, even though physical work has begun, which turns what would otherwise be a costly investigation into a controlled correction. When the project is later audited or disputed, the consultancy produces a signed, reversible design history rather than a folder of drawings whose version nobody is certain of.

A manufacturer selecting materials for a safety-critical component. A manufacturer is selecting the material for a component that has to carry load with an adequate safety margin, and the selection has to be defensible to a regulator and durable as a record. TITAN selects from specified materials grounded in the ASTM standards, verifies the safety margin the design requires rather than assuming it, and signs the selection and its justification into the OAR. If a later analysis prompts a change of material, the change is recorded with its declared inverse, so the path back is available and the history is complete. When the regulator asks the manufacturer to demonstrate that the material selection was made properly and with an adequate margin, the answer is a signed record showing the selection, the margin verification, and the standards it was made against, verifiable offline. The manufacturer's confidence in the component rests on a recorded engineering process rather than on institutional memory.

An asset owner managing a structure across its life. An asset owner is responsible for a structure over decades, during which it will be analysed, modified, monitored, and eventually assessed for life extension or decommissioning. TITAN gives the owner a continuous, signed engineering record: the original analysis against the Eurocode and BS and EN standards, every modification recorded with its declared inverse, and, through TITAN's integration with RAIDEN, a link to the real-

time monitoring of the asset's condition once built. When the time comes to assess the structure for continued service, the owner replays a tamper-evident chain spanning the asset's whole engineering life, showing every change and its reasoning, rather than reconstructing the history from incomplete records. The HSE Construction Design and Management duties that attach to the asset are easier to evidence because the design decisions that bear on safety are signed and retraceable. The asset's engineering provenance becomes a durable property rather than a fading institutional recollection.

Regulatory and standards relevance

TITAN's relevance to the regulatory and standards frame is the most direct of any brain in the subsystem, because engineering in the UK is a heavily standardised and statutorily regulated activity, and TITAN is built to work inside that frame rather than alongside it. Its knowledge base is the standards themselves: the Eurocode suite, the BS and EN standards, the AISC manual, and ASCE 7 for design; the ASTM standards for materials; the NEC4 suite and the ICE manual for contracting and construction; and, crucially, the HSE Construction Design and Management Regulations, which impose statutory duties on those who design and build. Because TITAN works to these standards, its outputs are expressed in the terms a regulator, a building-control authority, and a chartered engineer already use. Its compensating-inverse change tracking and signed design history are directly relevant to the duties under the CDM Regulations and to the evidentiary needs of any safety case or dispute, because both turn on the ability to show what was decided, when, by what reasoning, and that the record has not been altered. Through its integration with ZEUS, the regulatory and governance dimension of a design recommendation is carried by the brain that owns the law. TITAN does not certify a design as compliant, that judgement belongs to the responsible engineer and the relevant authority. It produces the signed, standards-grounded, reversible record on which compliance is demonstrated and defended.

What this brain does not do

TITAN is an engineering and infrastructure specialist, and its boundaries are well drawn. It does not derive the underlying physics from first principles, that is QUANTUM's domain, though it applies that physics through the shared computational-engineering surface. It does not monitor the real-time condition of what it designs, that is RAIDEN's work, though it integrates with RAIDEN to receive it. It does not render the legal and regulatory judgement on a design, that is ZEUS's

domain, though it integrates with ZEUS to carry it. It does not make irreversible changes silently: the compensating-inverse discipline means every design recommendation is recorded with its declared inverse, so an untracked, unrecoverable change is precisely what TITAN is built to prevent. And it does not substitute for the responsible engineer or the relevant authority; it produces analysis, plans, selections, and a signed reversible record, and the professional judgement and statutory sign-off remain with the people the law places them with.

Frequently asked questions

What is a declared inverse, and why does it matter in engineering? A declared inverse, from patent 14, is the compensating action that would undo a change, recorded together with the change itself. It matters in engineering because the cost of a change rises steeply once construction begins, and the ability to retrace exactly what was changed and how to undo it on paper, even after physical work has started, turns a potentially expensive investigation into a controlled correction. TITAN records every design recommendation with its inverse, so the path back is always available.

How does TITAN's signed record help in a dispute or safety investigation? TITAN signs every analysis, plan, selection, and change into the hash-linked OAR, including the declared inverse of each change, so the design history is both tamper-evident and reversible. In a dispute or investigation the owner can replay a signed chain showing exactly what was designed, what was changed, when, and on what reasoning, verifiable offline with only the public key. That is precisely the record such proceedings turn on and the hardest to assemble after the fact from drawings and emails.

Does TITAN replace the responsible engineer or building-control sign-off? No. TITAN performs analysis, capital planning, materials selection, and safety-margin verification against the relevant standards, and it produces a signed, reversible record. The professional judgement of the responsible engineer and the statutory sign-off of the relevant authority remain with the people the law places them with. TITAN makes their judgement better-evidenced and defensible; it does not assume their role.

Which standards does TITAN actually work to? TITAN's knowledge base is the working standards of UK engineering practice: the Eurocode structural-design suite, the BS and EN standards, the AISC steel manual, and ASCE 7 for design loads; the ASTM standards for materials; the NEC4 contract suite and the ICE manual for

construction; the HSE Construction Design and Management Regulations for statutory safety duties; and the UK Highway Design Manual and Engineering Toolbox for sector reference data. Its outputs are expressed in the terms a UK chartered engineer and regulator already use.

Chapter Five: KARP, the data and analytics brain

A figure you can prove came from a source

KARP is the data and analytics specialist of the Mickai cooperative, scoped in the catalogue to data, analytics, and business intelligence, with a one-line description that names its register precisely: tabular and structured data, synthesis, transformation, and signed analytical reporting. KARP is the brain of the spreadsheet, the dataframe, the query result, and the board report, the brain that turns rows and columns into the figures decisions are made on. It is the brain a finance function, an analytics team, or any data-driven decision-maker reaches for when the question is what does the data say, and the answer has to be both correct and provable.

The work KARP does is the work of a data analyst, raised onto a substrate that signs and traces every transformation. It generates spreadsheets and dataframes. It synthesises and executes queries against local stores. It produces analytical reporting. And it does all of this under a discipline the catalogue states with care: KARP produces signed transformations, so that a downstream consumer can prove that a particular figure came from a particular query against a particular signed source. That sentence is the whole value of the brain. The chronic problem of data analytics is not usually computing a number; it is being able to say, later and to a sceptic, where the number came from. A figure in a board report that nobody can trace back to its source is a figure that cannot be defended, and in a regulated or

audited environment an indefensible figure is a liability. KARP makes the lineage from source to figure a signed, replayable thing.

What KARP is responsible for

The Mickai catalogue gives KARP four declared responsibilities, and they describe the full path from raw data to defensible report.

Spreadsheet and dataframe generation. KARP generates spreadsheets and dataframes, the two fundamental containers of tabular analytical work. This is the bread and butter of data analysis, the construction of the structured tables in which data is held, transformed, and presented. KARP treats this as a first-class capability rather than an export format, because the spreadsheet or dataframe is where the analytical work actually lives, and producing it correctly, with the right structure and the right transformations, is the substance of the analyst's craft.

Query synthesis and execution against local stores. KARP synthesises queries and executes them against local stores. The phrase local stores is doing important work, and it echoes the perimeter discipline that runs through the whole subsystem: KARP queries data that lives on the operator's own machine, not data shipped to a vendor's warehouse. Query synthesis is the translation of an analytical question into the precise query that answers it, and execution against local stores means the answer is produced on the operator's hardware, against the operator's data, under the operator's control. The sensitive underlying rows never leave the perimeter.

Analytical reporting with signed provenance. KARP produces analytical reports, and the reports carry signed provenance. This is the responsibility that turns a number into a defensible figure. Every report KARP produces is bound, through the signed transformation chain, to the queries and the sources it was built from. A consumer of the report can prove, not assert, that a particular figure came from a particular query against a particular signed source. In an environment where a board report drives a capital decision and is later audited, this is the difference between a figure that survives scrutiny and a figure that collapses under it.

Row and column access control at the cell level. This is KARP's most distinctive responsibility, and it rests on the granular access-control primitive filed as patent 18. KARP's access control descends to the level of the individual cell, and it supports revocation. This is a far finer grain than the usual file-level or table-level permission. It means that within a single spreadsheet or dataframe, particular rows,

particular columns, or particular cells can be governed by their own access rules, and access can be revoked at that grain. For data that mixes sensitivities, a dataset where some columns are open and others are restricted, or where some rows pertain to parties with different access rights, this cell-level control is exactly what is needed to share the analysis without over-sharing the data.

What KARP reads, and what it works in

KARP's authority comes from the canonical statistical and economic data sources, and the catalogue names them. Its knowledge base spans the ONS Office for National Statistics datasets, Eurostat, the World Bank Open Data, the IMF Data Mapper, the FRED economic data, the HMRC open data, the Companies House register, the TIOBE and RedMonk technology indices, the ISO 8000 data-quality standards, and the OECD statistics. The shape of that list tells you what kind of analyst KARP is. It is grounded in authoritative, official statistics, the ONS, Eurostat, the World Bank, the IMF, the OECD, the sources a serious economic or business analysis cites, rather than in scraped or unverifiable data. The inclusion of the Companies House register and the HMRC open data grounds KARP in the UK's official corporate and tax data. And the presence of the ISO 8000 data-quality standards is pointed: KARP is not merely a brain that computes over data, it is a brain that holds a standard for what good data is, which is the foundation of trustworthy analytics. A figure built on poor-quality data is a poor figure however elegantly it is computed, and KARP's grounding in ISO 8000 reflects that the quality of the input is part of the defensibility of the output.

KARP works through a set of analytics tools cloned into the brain. It uses Codex, the shared sovereign plain-text graph knowledge surface, as its spine. On top of that it runs Prism, here in its role as a visual-analytics surface, where data is turned into the charts and visual summaries a report communicates through; Beacon, a full-text and structured-search surface, which is how KARP finds and retrieves the data it works over; Loom, here as an analytics directed-acyclic-graph scheduler, which sequences the steps of a data pipeline in dependency order; and Vellum, here as a board-ready report workspace, where the analysis is assembled into the document a board or a regulator will actually read. The toolset mirrors the analyst's workflow: search to find the data, the DAG scheduler to run the pipeline, the visual-analytics surface to present it, and the report workspace to deliver it. Loom is shared with JAXON, which is apt, because a data pipeline and a build pipeline are the same kind of object, a dependency-ordered sequence of steps, and both are scheduled the same way.

Where KARP sits in the subsystem

KARP's relationships reach beyond its own subsystem, because data and analytics feed decisions everywhere. Its catalogue entry names two. KARP feeds PALANTIR on strategic signals, because the strategic reasoning brain in the Intelligence and Defence subsystem needs structured data to reason over, and KARP is the brain that produces it in defensible form. And it feeds ZEUS on governance reporting, because the legal and governance brain needs analytical reports to discharge governance and regulatory obligations, and those reports have to carry the signed provenance that KARP provides. KARP is the subsystem's source of defensible structured data, and it feeds the brains, in other subsystems, whose work depends on data they can trust and trace. This is the cooperative pattern reaching across subsystem boundaries: a figure KARP produces, with its lineage signed, can become an input to a strategic assessment or a governance report without losing its provenance, because the signed chain follows the figure wherever it goes.

How every action is signed into the OAR

KARP's signing is the mechanism behind its central promise, that a figure can be proved to have come from a source, so it deserves a precise statement. Every transformation, every query execution, and every report KARP produces is signed under FIPS 204 ML-DSA-65 and appended to the Open Audit Record, hash-linked under SHA-3-512. The signed transformation chain is the thing that makes a figure provable: a number in a report is bound, through the chain, to the transformation that produced it, which is bound to the query that fed it, which is bound to the signed source the query ran against. A consumer of the report can walk that chain backward, from the figure to the source, and verify each link, offline, with only the public key. The hash linking guarantees the chain cannot be reordered or altered, so the lineage a consumer replays is provably the lineage that actually produced the figure.

The cell-level access control, from patent 18, interacts with this signing in a way worth drawing out. Because access control descends to the cell and supports revocation, the OAR records not only how a figure was produced but who was permitted to see what at each step, and revocation is itself a recorded, signed event. The consequence is an analytics audit trail of unusual completeness: it shows the lineage of every figure and the access governance around every cell, all tamper-evident and offline-verifiable. For a finance or governance function this is the difference between a report whose figures are asserted and a report whose figures are

proved, and between data access that is governed by policy on paper and data access that is governed by cryptography in fact.

Operator scenarios

A finance function preparing a board report. A finance function is preparing the figures a board will rely on for a capital decision, figures that will later be audited, and that must be both correct and defensible. With KARP the figures are produced by signed transformations against signed local sources, grounded where relevant in the ONS, HMRC, and Companies House data in KARP's knowledge base, and assembled into a board-ready report in Vellum. Every figure is bound through the signed chain to the query and the source it came from, so when the auditor asks where a particular number came from, the answer is a replayable chain rather than an analyst's recollection. The underlying rows, which may be commercially sensitive, never leave the operator's perimeter because KARP queries local stores. The board gets figures it can rely on, and the finance function gets figures it can defend, because the provenance is signed rather than asserted.

An analytics team sharing a mixed-sensitivity dataset. An analytics team needs to share an analysis built on a dataset that mixes sensitivities, some columns open, some restricted, some rows pertaining to parties with different access rights. The usual file-level or table-level permissions are too coarse: they force a choice between sharing the whole dataset or none of it. KARP's cell-level access control, from patent 18, lets the team govern access at the grain the data actually requires, so the analysis can be shared while the restricted cells stay restricted, and access can be revoked at that same grain if circumstances change. Every access decision and every revocation is signed into the OAR, so the team can demonstrate, to a data-protection or governance function, exactly who was permitted to see what and when that changed. The analysis is shared without the over-sharing that coarse permissions would force.

A governance function evidencing a regulatory report. A governance function has to produce a regulatory report whose figures must be evidenced to the regulator's satisfaction, and KARP feeds exactly this work, since its catalogue entry names ZEUS, the governance brain, as a consumer. KARP produces the analytical figures with signed provenance, grounded in authoritative statistics and held to the ISO 8000 data-quality standard, and the signed transformation chain binds each figure to its source. When the report reaches ZEUS for the governance and regulatory framing, the figures it carries already have their lineage signed, so the

governance opinion rests on data that is itself provable. When the regulator examines the report, the function can replay the lineage of any figure offline, demonstrating not merely that the number is what it is but that it came from where it is claimed to have come from. The report's credibility rests on cryptographic provenance rather than on the regulator's willingness to take the figures on trust.

Regulatory and standards relevance

KARP's relevance to the regulatory and standards frame runs along three lines. The first is data quality: KARP's grounding in the ISO 8000 data-quality standards means it holds an explicit standard for what good data is, which matters because the defensibility of an analytical figure begins with the quality of its inputs, and regulators increasingly expect organisations to evidence data quality rather than assume it. The second is provenance and auditability: the signed transformation chain that binds every figure to its source is precisely the kind of evidence that financial, governance, and data-protection regimes ask for when they require that reported figures be traceable and that the integrity of reporting be demonstrable. The third is access governance: the cell-level access control and revocation, from patent 18, speak directly to data-protection and confidentiality obligations, because they let an organisation govern and evidence access to sensitive data at a fine grain rather than relying on coarse, hard-to-evidence file permissions. Because the signing pipeline rests on FIPS 204 ML-DSA-65 and SHA-3-512, the integrity claims are checkable cryptographic claims. As with the rest of the subsystem, KARP does not certify a report as compliant, that judgement belongs to the responsible function and the relevant authority. It produces the signed, quality-grounded, access-governed figures on which compliant reporting is built and defended.

What this brain does not do

KARP is a data and analytics specialist, and it is disciplined about its scope. It does not perform the strategic reasoning that belongs to PALANTIR or the legal and governance judgement that belongs to ZEUS, though it feeds both with defensible data. It does not ship the operator's underlying rows to any external warehouse, because it queries local stores, so a reading of KARP that imagines cloud analytics has misunderstood the brain. It does not produce figures whose lineage is unrecorded, because the signed transformation chain is how the brain operates, which means KARP cannot produce an untraceable figure even if asked to. And it does not govern data access at a coarse grain when a fine one is required: the cell-level access control exists precisely so that access can be governed at the grain the

data demands rather than forced to the level of the whole file. KARP's restraint is the point: a brain whose value is provable data cannot also be a brain that produces figures nobody can trace, because the untraceable figure is exactly what the brain is built to refuse.

Frequently asked questions

How can a consumer prove where a figure in a KARP report came from? Every transformation, query, and report KARP produces is signed and hash-linked in the OAR, so a figure is bound through the chain to the transformation that produced it, the query that fed it, and the signed source the query ran against. A consumer can walk that chain backward from the figure to the source and verify each link offline with only the public key. The provenance is proved, not asserted, which is the difference between a figure that survives an audit and one that does not.

What is cell-level access control, and when does it matter? Cell-level access control, from patent 18, means KARP's access governance descends to the individual cell and supports revocation, rather than stopping at the file or table. It matters whenever a dataset mixes sensitivities, some columns open and others restricted, or rows pertaining to parties with different rights, because it lets the analysis be shared without over-sharing the data, and access can be revoked at that same fine grain. Every access decision and revocation is signed into the OAR.

Does KARP send our underlying data anywhere? No. KARP synthesises and executes queries against local stores, on the operator's own hardware. The sensitive underlying rows never leave the operator's perimeter. KARP produces the figures and reports on the operator's machine, against the operator's data, under the operator's keys, which is what lets it be used on commercially sensitive or regulated data that could not be shipped to a cloud analytics service.

Why does KARP's knowledge base include a data-quality standard? Because the defensibility of an analytical figure begins with the quality of its inputs, and a figure built on poor data is a poor figure however well it is computed. KARP's grounding in the ISO 8000 data-quality standards reflects that it holds an explicit standard for what good data is, which is increasingly what regulators expect organisations to evidence. Data quality is, for KARP, part of the provenance that makes a figure defensible.

Closing: how the five brains cooperate, and the substrate beneath them

The five brains of the Science and Engineering subsystem are not five separate tools that happen to share a logo. They are five specialists in one cooperative, and the value of the subsystem is in how they work together as much as in what each does alone. It is worth tracing the cooperation directly, because it is the clearest way to see what a Sovereign Intelligence Operating System is, as opposed to a collection of AI features.

Consider a single piece of consequential technical work, and watch it move through the subsystem. A physical result is needed, so QUANTUM derives it, with the working signed and the uncertainty carried honestly. The result has to become a numerical method that runs, so QUANTUM hands the implementation to JAXON, which writes and tests the code on-device, the source never leaving the machine, every patch signed. The result has an engineering application, so TITAN takes it into a structural analysis and a design, recording every change with its declared inverse so the path back is always available. The thing TITAN designs, once built, generates real-time signals about its condition, so RAIDEN watches them to a deadline, raising signed, provenance-bearing alerts if a condition develops. And the data the whole exercise generates, the measurements, the costs, the performance, becomes structured analytical reporting in KARP, with every figure bound through a signed chain to its source. Five brains, five specialisms, one continuous chain of signed work, from the physics to the code to the structure to the monitoring to the report.

The cooperation is not metaphorical. It is wired into the catalogue. QUANTUM coordinates with JAXON on numerical implementation and with TITAN on engineering application. TITAN integrates with QUANTUM on the underlying physics and with RAIDEN on real-time monitoring. RAIDEN's output stream feeds TITAN on infrastructure response. KARP feeds PALANTIR and ZEUS in the Intelligence and Defence subsystem. The handoffs cross subsystem boundaries when the work demands it, and when they do, the signed chain follows the work across, so that a figure KARP produces can become an input to a governance report in ZEUS without losing its provenance, and a real-time signal RAIDEN detects can become a humanitarian response in SALVATOR without losing its timeline. The subsystem boundaries organise the specialists; they do not wall them off.

Underneath all five brains, and underneath every other brain in the cooperative, sits the same substrate, and it is the substrate, more than any individual capability, that

makes Mickai what it is. Three primitives are worth restating once more, because they are the foundation the whole account rests on.

The first is the signature. Every consequential action any of these five brains takes is signed at the moment of commit under FIPS 204 ML-DSA-65, the NIST post-quantum digital-signature standard finalised in 2024. The signature is produced under a key the operator holds in hardware, not a key a vendor holds in a cloud. It is post-quantum-secure today, ahead of the NCSC migration deadlines, which matters for any organisation whose records have to remain verifiable into a future where the cryptography of today may not hold. A signature does not assert that an output is correct. It asserts that this exact output came from this exact operator's system at this exact time and has not changed since, which is the ground that any argument about correctness has to stand on.

The second is the chain. The signed records append to the Open Audit Record, a hash-linked chain under SHA-3-512. Each entry carries the hash of the entry before it, so the chain is tamper-evident: an entry cannot be inserted, removed, reordered, or back-dated without breaking the link, and the break is visible to anyone who replays it. The OAR is not a log the operator is asked to trust. It is a cryptographic structure whose integrity can be checked. For the science and engineering domain, where the timeline of a derivation, a design change, a grid event, or a data transformation is frequently the thing under examination, a chain whose ordering is cryptographically fixed is precisely the record that ordinary logs cannot provide.

The third is externalisation. The audit chain lives under the operator's key in an open format, and the verifier that checks it is browser-resident and offline. This is what Mickai means by trust-domain externalisation: the chain does not depend on the AI vendor being trusted, being online, or even being in business. The operator, a regulator, a peer reviewer, an auditor, or any third party with the public key can replay the same chain and reach the same deterministic verdict, independently and offline. The verifier does not phone home. It takes the chain and the key and it returns a verdict. The trust that a conventional AI deployment asks the buyer to place in the vendor is, in the Mickai substrate, externalised into a cryptographic structure the buyer can check for themselves. That is the difference between being told a system is trustworthy and being able to verify that it is.

These three primitives are not described here as marketing properties. They are the substance of the patent corpus filed at the UK Intellectual Property Office under the GB2607309.8 to GB2611702.8 family, named inventor Micky Irons, and they are referenced throughout this ebook only where the brains' own catalogue entries

reference them: the pre-commit simulation that JAXON uses, the hardware-bound alert key that RAIDEN signs with, the compensating inverse that TITAN records, the cell-level access control that KARP enforces. The filings are filings, not grants, and they are described here in terms of what they contain and what the brains do with them.

A procurement note

For the buyer who has to put this in front of a technical-assurance committee, a procurement function, or a board, it is worth setting out plainly what the Science and Engineering subsystem is and is not, because the worst outcome in this domain is a system bought on a claim it cannot support.

What it is: a sovereign substrate that brings frontier technical AI capability inside the operator's perimeter, under the operator's keys, with a complete, signed, hash-linked, offline-verifiable record of everything it does. The capability is real and it is broad, spanning on-device software engineering, real-time signals and alerting, hard-sciences derivation, engineering and infrastructure design, and data analytics. The provenance is cryptographic, not contractual. The data stays on the operator's machine, not in a vendor's cloud. And the audit chain can be verified by a third party offline, with only a public key, which means the operator's ability to demonstrate control does not depend on the vendor's cooperation or continued existence.

What it is not: it is not a system that certifies its own outputs as correct or compliant. None of these five brains adjudicates correctness; they make the working reproducible and the lineage provable so that the responsible engineer, the peer reviewer, the auditor, and the relevant authority can do their jobs with better evidence. JAXON does not certify a codebase, it produces a provable record of how the code was made. QUANTUM does not declare a result true, it makes the derivation replayable so peer review can. TITAN does not sign off a design, it produces a signed, reversible record on which the responsible engineer's sign-off rests. KARP does not certify a report compliant, it produces figures whose lineage is signed. The professional and statutory judgements remain exactly where the law and the profession place them. The substrate makes those judgements better-evidenced; it does not assume them.

The questions a technical-assurance committee should ask are the questions this subsystem is built to answer. What algorithms underpin the signing and the hash-linking? FIPS 204 ML-DSA-65 and SHA-3-512, published standards a committee can check. Where does the data go? Onto the operator's own hardware and nowhere

else, because the brains read repositories, query local stores, and compute on-device by construction. Can a third party verify the record without trusting the vendor? Yes, offline, with only the public key, through a browser-resident verifier that returns a deterministic verdict. Is the post-quantum posture ahead of the migration timeline? Yes, the signature is post-quantum-secure today. Who holds the keys? The operator, in hardware. These are concrete, checkable answers, and a procurement process that asks them will find that the subsystem was designed around exactly these questions rather than around the convenience of a cloud deployment.

There is a final point worth making to the buyer, and it is about the long life of technical work. The work this subsystem does, code, derivations, designs, alerts, reports, has consequences that outlast the moment it is produced. A structural design is challenged years later. A derivation is checked by a referee. A grid event is investigated after the fact. A board figure is audited. In every case the value of a signed, hash-linked, offline-verifiable record is not realised on the day the work is done. It is realised on the day the work is questioned, which may be years away and may involve people who were not there and have no reason to trust anyone's recollection. The Science and Engineering subsystem is built for that day. It records the working as it happens, in a form that survives transmission to a sceptic, so that when the question comes, the answer is a chain that can be replayed rather than a story that has to be reconstructed. That is the property the technical professions have always needed and rarely had, and it is the property the substrate was built to provide.

Mickai is the British Sovereign Intelligence Operating System. The Science and Engineering subsystem is five brains of it: JAXON, RAIDEN, QUANTUM, TITAN, and KARP, each a specialist in its domain, each signing every consequential action into an open, tamper-evident chain the operator holds and anyone can verify. It is held privately by its founder, Micky Irons, and its substrate primitives are filed at the UK Intellectual Property Office. The capability is for the engineer at the desk, the researcher at the bench, the operator in the control room, and the analyst preparing the figures. The provenance is for everyone who comes after, asking how the work was done, and needing to be shown rather than told.

A glossary of the substrate

Sovereign Intelligence Operating System (SIOS)

Frontier-class AI that runs on the operator's own hardware, signs every action it takes, and produces a record any third party can verify offline.

Brain

A specialist unit of the Mickai SIOS, scoped to a domain or a cognitive function, signed and audited like every other action in the system.

Open Audit Record (OAR)

The signed, hash-linked record of every action the SIOS takes, designed to be verified offline by anyone holding the operator's public key.

FIPS 204 ML-DSA-65

The United States NIST post-quantum digital signature standard, used to sign every action so the audit chain survives a future quantum adversary.

SHA-3-512

The hash function used to link each audit record to its predecessor, so the chain cannot be altered retrospectively without detection.

Trust-domain externalisation

The pattern in which the record of an action is held under the operator's key in an open format, so the operator, a regulator, and any third party can verify it without the vendor.

Operator-held keys

The cryptographic keys that sign the audit chain are held by the operator in their own hardware, not by the AI vendor.

Browser-resident verifier

A static, offline verifier that loads an audit chain in a browser, checks every signature and hash link, and returns a deterministic verdict with no server call.

Poseidon

The operator-personalised sovereign silicon substrate beneath the Mickai SIOS, the hardware root of trust the keys are bound to.

Post-quantum

Cryptography that remains secure against an adversary equipped with a cryptographically relevant quantum computer.

Deterministic routing

The property by which the same request, in the same context, under the same policy always routes to the same brains in the same order, so the audit chain is replayable.

Pre-commit dry run

A simulation of a high-impact action, rendered as a difference against the target state, that the operator reviews before the action commits.

Quorum

The pattern in which a high-stakes decision is dispatched to several independent brains, and no result is signed unless they agree within a defined threshold.

Air gap

An operating mode in which the SIOS runs with no network connection, with bootstrap and attestation handled entirely on operator hardware.

Revocation

The withdrawal of a previously granted authority, recorded as a signed tombstone that downstream verifiers honour.

CBOR

A deterministic binary encoding used for audit records, producing a single canonical byte representation for any record.

The Fifty Brains

This volume is one of five in The Fifty Brains, a series on the brains of the Mickai Sovereign Intelligence Operating System.

Mickai is the British Sovereign Intelligence Operating System. It runs frontier-class AI on the operator's own hardware, signs every action under the operator's own post-quantum key, and produces the Open Audit Record that anyone can verify offline. The full brain catalogue is at mickai.co.uk/brains.

MICKAI LTD · COMPANIES HOUSE 17166618 · TRADE MARK UK00004373277 · MICKAI.CO.UK

Further reading

The wider Mickai corpus is at mickai.co.uk/ebooks and mickai.co.uk/articles. Companion technical volumes include:

The Audit Substrate Under Every AI Agent

The Twenty-Five Brain Architecture

Trust-Domain Externalisation, An Architectural Pattern for Sovereign AI

The UK Procurement Checklist for Sovereign AI

Post-Quantum Audit for Critical National Infrastructure

Every action the Mickai SIOS takes is signed under the operator's own post-quantum key and written into the Open Audit Record, verifiable offline by anyone. Sovereignty by proof, not by promise.