MICKAI™

MICKAI EBOOK SERIES · No. 17

Air-Gapped Intelligence.

Intelligence for the places the cloud cannot reach: contested environments, the degraded edge, and the systems that must keep thinking with the cable pulled.

AUTHOR

Micky Irons

Founder and named inventor, Mickai LTD.

19 June 2026 · v1 · mickai.co.uk

Mickai LTD · Companies House 17166618 · press@mickai.co.uk · mickai.co.uk
UK IPO register, named inventor Mickarle Wagstaff-Irons · Trade mark UK00004373277

TABLE OF CONTENTS

Contents

FOREWORD

A note from the author

I have spent years now building for the moment the cable gets pulled. Most of the industry has spent those same years assuming the cable never will. Those are two very different engineering cultures, and the gap between them is the subject of this book. I am Micky Irons, and I founded Mickai to build a Sovereign Intelligence Operating System, a SIOS, that keeps thinking on the operator's own hardware whether the link to the wider world is healthy, degraded, or gone entirely. This is not a thought experiment for me. It is the core design constraint of everything we have filed and everything we have shipped.

The phrase air-gapped usually arrives wrapped in fear. We picture a sealed room, a classified network, a machine that touches nothing. I want to reclaim the term as something quieter and more universal. An air gap is just the honest admission that connectivity is a privilege, not a guarantee. A lifeboat has an air gap. A submarine has one. A clinic in a valley after the storm has one. A soldier in a jammed corridor of spectrum has one. The cloud cannot reach any of them, and yet each of them still needs to think, decide, and remember. This book answers one question: how do you build intelligence that does not fall silent the instant it is alone.

What follows is the reasoning behind our choices, written plainly. I will show you why offline-first is an architecture and not a marketing word, why partition tolerance forces real decisions you cannot defer, how we seal every consequential action into a tamper-evident record even when nothing can be sent anywhere, and how all of it reconciles cleanly to Pantheon, our sovereign Bitcoin-anchored Layer 1, the moment a link returns. I have kept the prose free of hype and full of mechanism. Where there is a hard trade-off, I have named it rather than buried it.

I wrote this in the first person because these are my decisions and I will stand behind them. If you build systems for hostile, remote, or simply unreliable places, I hope you finish with a sharper sense of what is genuinely possible at the edge, and a lower tolerance for any vendor who tells you their intelligence needs their datacentre to function. The places the cloud cannot reach are not edge cases. They are most of the world, most of the time.

Micky Irons

Founder and named inventor, Mickai LTD · 19 June 2026

PART I · THE PROBLEM

Why connectivity-dependent intelligence fails exactly where it matters most.

1. The cloud is a place, and it has edges

The modern AI stack carries a hidden assumption so deep that most builders never name it: that there is always a route to a datacentre. Inference happens somewhere far away, on hardware you do not own, reached over a network you do not control, and the answer returns along that same fragile thread. When the thread holds, the illusion is seamless. The model feels local, instant, omnipresent. But the cloud is not an abstraction floating above the world. It is a specific set of buildings in specific jurisdictions, joined by specific cables and specific spectrum, and every one of those things has an edge where it stops.

I have watched capable teams discover that edge the hard way. They build a brilliant assistant, demonstrate it under perfect conditions, then ship it into a clinic, a vessel, a forward position, or a rural site where the bandwidth is a rumour. The moment the link degrades, the intelligence does not degrade with it. It vanishes. The spinner turns, the request times out, and the operator who needed a decision in that exact second is handed a blank screen and the quiet humiliation of a tool that worked fine in the demo and failed in the field.

> **Intelligence that lives in someone else's building is not your intelligence. It is a service you are renting, and the lease is cancelled the instant the link drops.**

This is the first thing to understand clearly. Connectivity-dependent AI is not slightly worse in a contested environment. It is categorically absent. There is no partial credit. A model that needs the cloud and cannot reach the cloud is, for every operational purpose, a model that does not exist. And the environments where you most need a sharp, fast, private decision are precisely the environments where the cloud is least likely to be there: the contested, the remote, the degraded, the deliberately denied.

So we begin by relocating the intelligence. Not as a feature, not as an offline mode bolted on at the end, but as the foundational decision from which everything else follows. The SIOS runs on the operator's own hardware. The fifty specialised brains live where the operator lives. The cloud, when present, is a convenience to reconcile with, never a dependency to survive on. Make that single inversion and a great many problems that were previously impossible become tractable.

2. Contested, degraded, denied: a taxonomy of broken links

Not all connectivity failures are the same, and treating them as one undifferentiated outage leads to lazy engineering. I find it useful to separate three conditions, because each demands a different response from the system. The first is the contested environment, where the link is under active attack. Spectrum is jammed, routes are poisoned, an adversary is degrading your communications on purpose. The failure here is hostile, intermittent, and unpredictable. You may get three seconds of clean link, then ninety seconds of nothing, then a burst of corrupted packets engineered to make you trust bad data.

The degraded edge

The second condition is the degraded edge, where nobody is attacking you but the physics are simply against you. A research station, a deep mine, a ship far from shore, a disaster zone where the towers fell. The link is not hostile, just thin, expensive, high-latency, and unreliable. You might get a satellite window for ten minutes an hour, or a few kilobits per second when you have anything at all. The system has to make every byte count and assume the window will close without warning.

The denied environment

The third condition is the deliberately denied environment, the true air gap, where there is no link by design. Classified networks, sealed facilities, systems that must never touch an outside network because the cost of compromise is catastrophic. There is no outage to wait out. The disconnection is permanent and intentional, and any architecture that treats connectivity as the normal state and disconnection as the exception is fundamentally unsuited to the work.

This taxonomy matters because a system built only for one of these conditions tends to fail the others. Build for the denied environment alone and you may neglect reconciliation entirely, assuming the link never returns. Build for the degraded edge alone and you may squander your thin window on the wrong traffic. Build for the contested environment alone and you may grow paranoid about data you could safely trust. The SIOS holds all three in mind at once, because the same operator may move through all three in a single mission, and the system must never notice the difference where it counts. It just keeps thinking.

The Mickai pantheon.

3. The honest cost of pretending the link is always there

There is a comfortable lie at the heart of most cloud-first design, and it is worth saying out loud. The lie is that disconnection is rare, brief, and exceptional, so it is acceptable to handle it badly. Engineers internalise this because in a well-served city it is nearly true. Their own link almost never drops. They build for their own conditions and ship those conditions to people who do not share them. The result is software that is quietly hostile to anyone living outside the bandwidth-rich core of the connected world.

Count the real cost. When a connectivity-dependent system fails at the edge, the operator does not simply lose a feature. They lose the audit trail, because the logging went to a server they cannot reach. They lose the decision, because the model that would have made it is unreachable. They lose trust, because the tool has failed them at the exact moment they relied on it, and people do not forgive tools that fail under pressure. In the worst cases they lose the data they generated during the outage entirely, because nothing local was designed to hold it.

There is a deeper cost too, harder to measure but more corrosive. Connectivity dependence is a sovereignty problem dressed up as a convenience. If your intelligence lives in someone else's datacentre, then someone else can switch it off, throttle it, inspect it, subpoena it, or price it out of your reach. The air gap is not only a resilience strategy. It is the physical guarantee that your thinking remains yours. A jammer cannot read what never left the building. A subpoena cannot reach what was never uploaded. The same architecture that survives the outage also delivers the privacy.

> **Design for the worst link your operator will ever see, and the good link becomes a bonus rather than a crutch.**

So I do not treat disconnection as an exception to be apologised for. I treat it as the design centre. The SIOS assumes the link is absent and is pleasantly surprised when it is present. That single change of posture, from connected-by-default to disconnected-by-default, is the most important architectural decision in this book, and every chapter that follows is a consequence of it.

PART II · THE ARCHITECTURE

Offline-first as a discipline, and the trade-offs partition tolerance forces you to face.

4. Offline-first is an architecture, not a fallback

The phrase offline-first gets used loosely, so let me be precise. Offline-first is not a degraded mode you switch into when the network drops. It is the primary mode, the one the system is built around, with online treated as an enhancement layered on top. The difference is not cosmetic. In a fallback architecture the offline path is the afterthought, the poorly tested branch nobody exercises until it fails. In an offline-first architecture the offline path is the main road and the online path is the slip road.

Concretely, this means the model weights live on the operator's hardware. The fifty specialised brains of the SIOS are present locally and run locally, fully offline-capable, so inference never requires a round trip to anywhere. It means the data the operator generates is written first to local durable storage, not streamed to a remote endpoint that may not answer. It means the interface is built to function with zero network, showing real state rather than a perpetual loading spinner. Every component is designed to be complete and correct on its own, with synchronisation treated as a separate concern that happens when it can.

Local by default, remote by exception

This inversion has a cost, and I will not pretend otherwise. You carry the weights with you, which means you provision real hardware and size it honestly to the work. You cannot lean on an infinite remote datacentre for the heaviest models. You make deliberate choices about which capabilities run on the device in front of you and which need more silicon than the operator is carrying. But this cost buys something no cloud architecture can offer: certainty. The intelligence is there because it is physically there, not because a chain of intermediaries between you and a distant building all happened to be working at once.

The discipline shows up in a thousand small decisions. Does this feature assume a server response? Then it is not offline-first, and it must be redesigned. Does this log write hit the network before it hits local disk? Then a power loss during an outage destroys it, and that is unacceptable. Does the user see lesser behaviour when disconnected? Then we have built a fallback and called it offline-first, and we have lied to ourselves. The test is simple and unforgiving. Pull the cable in the demo. If the demo continues as though nothing happened, you have built offline-first. If it stumbles, you have built a fallback wearing the word offline-first as a costume.

The Mickai pantheon.

5. Partition tolerance: the choice you cannot avoid

Once you commit to offline-first, you collide immediately with one of the oldest hard truths in distributed systems. When a network partitions, when the link between two parts of a system breaks, you cannot have both perfect consistency and continued availability at once. You must choose. Either the system refuses to act until it can confirm it agrees with the rest of the world, which preserves consistency but sacrifices availability, or it keeps acting on its local knowledge and reconciles later, which preserves availability but accepts temporary inconsistency. There is no third option that escapes the trade-off. The mathematics does not permit it.

> **In a partition you choose between a system that stops to stay correct and a system that keeps going and gets correct later. At the contested edge, only one of those is acceptable.**

For the environments this book is about, the choice makes itself. A clinician with no link cannot wait for a distant database to confirm consensus before recording a treatment. A team in a denied facility cannot pause every decision until the air gap is bridged, because it never will be. Availability under partition is not a preference for us. It is the entire point. So the SIOS chooses to remain available, to keep thinking and keep recording through the partition, and accepts that the global picture will be temporarily divergent. The discipline then is making that divergence safe, bounded, and fully reconcilable, rather than chaotic.

Making divergence safe

Choosing availability does not mean choosing chaos. It means engineering the local behaviour so that everything done during the partition can be cleanly merged back later without loss, without ambiguity, and without trusting clocks you cannot verify. This is where the design earns its keep. Each action taken during the outage is recorded in a form that carries its own ordering, its own provenance, and its own integrity guarantee, so that when the link returns there is never a guessing game about what happened, in what order, and whether it can be trusted. The partition becomes a temporary local branch of reality that the system knows how to fold back into the trunk.

The brains themselves are partition-aware in their reasoning. A brain that knows it is operating without a link does not silently present stale external data as current. It flags the boundary of its own knowledge, marks what it could not refresh, and reasons honestly within those limits. Sovereign intelligence is not intelligence that pretends to an omniscience it does not have. It is intelligence that knows the shape of its own isolation and is candid about it, which paradoxically makes it far more trustworthy than a connected system that quietly serves you stale answers without ever admitting the link was down.

6. Sizing the silicon: honest hardware for the edge

Offline-first turns an abstract question into a concrete one. If the intelligence runs on the operator's own hardware, then the hardware in front of the operator sets the ceiling on what that intelligence can be. There is no escaping this with a clever API call to somewhere bigger. I find that constraint clarifying rather than limiting, because it forces an honesty cloud architectures let you avoid. You cannot hand-wave the compute. You have to look at the actual device and ask what it can actually run.

The answer is a spectrum, and the SIOS detects where on that spectrum it is running and scales accordingly. On a modest edge device, the smaller specialised models run comfortably and deliver real value within their domain. On a serious workstation-class machine, larger models in the higher parameter classes become reachable through careful memory management and hybrid execution across the available processors. On a flagship edge server, the full weight of the fleet can be brought to bear. The same SIOS spans all of these, detecting the hardware at runtime, sealing a profile of what it found, and presenting capabilities that match reality rather than promising what the silicon cannot deliver.

Never omit, always disclose

The principle I hold to is that we build every capability, even the ones a given machine cannot run, and we are honest about it. A feature that exceeds the current hardware does not silently disappear from the interface. It appears, clearly marked as unavailable on the present configuration, with guidance on what would be required to enable it. This matters because the operator deserves to know the full shape of the system, not a quietly truncated version edited down to whatever happens to fit. Omission is a kind of dishonesty. Disclosure with guidance is respect.

This is also why I am precise about what runs where and never overstate it. A large model that needs heavy offload across multiple processors is not the same as that model served instantly from abundant memory, and pretending otherwise would be exactly the hype I refuse to traffic in. The operator in a contested environment is betting on the system being truthful about its own limits.

Hardware-honest design is not a marketing posture. It is a safety property. A tool that lies about what it can do under load is a tool that gets people hurt when the load arrives.

The Mickai pantheon.

PART III · SEALING AND SURVIVING THE OUTAGE

Tamper-evident records, durable queues, and intelligence that holds the line alone.

7. The Open Audit Record: sealing what happens in the dark

Here is a question that exposes most edge architectures immediately. When your system takes a consequential action during a total communications blackout, where does the proof go? In a cloud-first design the answer is uncomfortable: nowhere, or into a local file anyone could later edit, because the real audit log lived on a server you could not reach. The outage that took your connectivity also took your accountability. That is a failure I was not willing to accept, because the environments this book is about are exactly the environments where accountability matters most and is hardest to preserve.

So in the SIOS, every consequential action is sealed into an Open Audit Record at the moment it happens, locally, with no network required. The record captures what was done, by which brain, on what basis, at what point in an ordered chain. Crucially, it is cryptographically signed the instant it is created, on the operator's own hardware, so its integrity does not depend on anything reaching anywhere. The proof is generated in the dark, sealed in the dark, and held in the dark, ready to be presented intact whenever there is finally something to present it to.

> **Accountability that depends on connectivity is no accountability at all. The record must be sealed where the action happens, the instant it happens, with nothing else in the loop.**

The signatures are post-quantum by design, using ML-DSA-65 under the FIPS 204 standard. I made that choice deliberately and early, because an audit record is a long-lived object. A record sealed today may need to prove its integrity a decade or more from now, and the cryptography that protects it must still be standing when that day comes. Signing with a scheme already standardised against the threat of quantum attack means the seal made in a denied facility today is not quietly rotting toward forgeability. It is built to outlast the machines that might one day try to break it.

The word open in Open Audit Record is not decoration. The format is designed to be independently verifiable, so a record's integrity can be checked by anyone with the public verification material and the open specification, not only by us and not only by the machine that made it. A sealed record you cannot independently verify is just a claim. A sealed record anyone can verify is evidence. For systems operating in the places the cloud cannot reach, often the very places where trust is most contested, that distinction is the whole game.

8. Durable queues: holding the truth until the link returns

Sealing a record is half the problem. The other half is holding it safely through an outage of unknown length and getting it where it needs to go the moment a path opens. This is the job of the durable queue, and it is far more subtle than a simple list of things to send later. A naive outbox loses data on power failure, sends things in the wrong order, sends them twice, or quietly drops the ones that do not fit in the next thin window. At the contested edge, every one of those failures is a real loss with real consequences.

Durable means it survives the power cut

Durable has a strict meaning here. A record entering the queue is committed to storage that survives a sudden loss of power before the operation is acknowledged. If the lights go out the instant after the system seals a record, that record is still there when the lights come back. This sounds obvious and is routinely got wrong, because writing to memory or to an unflushed buffer feels durable in the demo and proves catastrophically temporary in the field. The queue is engineered so an acknowledged record is a record that will be there after the worst happens, not merely a record that was there a moment ago.

Ordered, idempotent, and patient

The queue preserves the order in which actions occurred, because the sequence of events is itself part of the truth being recorded. It is idempotent on delivery, so a window that closes mid-transmission and forces a retry does not result in the same record being counted twice when the link finally holds. And it is patient without limit. Whether the partition lasts ten minutes or ten months, whether it is a satellite gap or a permanent air gap bridged only by a courier carrying physical media, the queue holds its contents intact and ordered, waiting, never assuming the wait will be short.

This patience is what lets the same architecture serve the degraded edge and the truly denied environment without modification. In the degraded case the queue drains quickly when the satellite window opens. In the denied case it may drain only through a deliberate, controlled transfer across the air gap, perhaps onto removable media physically carried to a connected system. The queue does not care which it is. It seals, it orders, it holds, and it hands over its contents intact whenever a path of any kind appears. The mechanism is indifferent to how long the dark lasts, which is exactly the property the dark demands.

The Mickai pantheon.

9. Thinking alone: the brains under isolation

A system that merely records during an outage is a logbook. A system that keeps reasoning during an outage is an intelligence. The harder and more valuable property is that the fifty brains continue to think while isolated, delivering real analysis and real decisions on local hardware with no link to anywhere. This is the payoff of carrying the weights with you. The operator does not lose their analyst, their adviser, their domain specialist when the cable is pulled. They keep all fifty, running on the silicon in front of them.

Thinking alone well requires a particular honesty in the reasoning itself. A brain operating under isolation must distinguish sharply between what it knows from its own resident knowledge and what it would normally refresh from outside but currently cannot. It marks the boundary explicitly. It does not present a stale external figure as though it were freshly confirmed. It reasons confidently within the perimeter of what it can actually verify locally, and it is candid about where that perimeter lies. This candour is not a weakness. It is the single most important property that makes an isolated intelligence trustworthy.

> **An isolated brain that admits the edge of its own knowledge is worth more than a connected one that hides it.**

The custodial brains earn their place precisely during long isolation. One watches over the freshness of knowledge, tracking what has gone stale and what still holds, so the system ages gracefully rather than silently rotting. Another watches over the health of the fleet itself, detecting and repairing degradation in the running system, because in a denied environment there is no engineer to call.

Self-maintenance is not a luxury at the edge. It is the difference between a system that survives a long isolation intact and one that quietly decays into unreliability while nobody is watching.

And throughout all of this, the sealing never stops. Every consequential conclusion the brains reach, every action they take during the isolation, is folded into the same Open Audit Record chain and held in the same durable queue. The intelligence and the accountability are not separate systems where one survives the outage and the other does not. They are a single fabric. The brain thinks, the action is sealed, the record is queued, and all of it waits together in the dark for the moment the world becomes reachable again. The system does not merely endure the outage. It works through it, fully and accountably, as though the dark were just another operating condition, which for these environments it is.

PART IV · RECONCILIATION AND THE SOVEREIGN LEDGER

Folding the dark back into the light: reconciling to Pantheon when the link returns.

10. The return of the link: reconciliation as a first-class event

Everything so far has prepared for a single moment: the return of the link. After a partition of any length a path opens, and now the local branch of reality that accumulated in the dark must be folded back into the shared trunk. I treat this reconciliation as a first-class event in the system, designed and tested as carefully as the outage itself, because a great deal of edge software handles the disconnection beautifully and then mishandles the reconnection catastrophically. The reunion is where the data gets corrupted, duplicated, or lost, if you have not engineered it with the same seriousness as the separation.

Reconciliation in the SIOS is the controlled draining of the durable queue into the shared record, in order, with integrity preserved end to end. Because every queued record is already sealed and signed, the reconciling system does not have to take anything on trust. It verifies each record's signature, confirms its place in the ordered chain, and folds it into the shared history with full assurance that nothing was altered during the dark and nothing is being smuggled in now. The seal made in isolation is the very thing that makes the reunion safe. Without it, reconciliation would be an act of faith. With it, reconciliation is an act of verification.

Idempotent, ordered, verifiable

Three properties make reconciliation trustworthy. It is idempotent, so a reconnection that fails partway and is retried does not double-count anything that already landed. It is ordered, so the sequence of events from the dark is preserved exactly in the shared record rather than scrambled by the accidents of transmission. And it is verifiable, so every record's integrity is independently checked as it lands, not assumed. A reconciliation that lacks any one of these three will eventually corrupt the very history it was meant to preserve, usually at the worst possible moment and usually in silence.

Reconciliation also handles the harder case of genuine divergence, where the same world was acted upon from more than one isolated branch. Because each branch carries its own sealed, ordered provenance, the merge is not a guessing game over unreliable timestamps. It is a principled fold of two fully attested histories, each independently verifiable, into a single coherent record that loses nothing and invents nothing. The dark may have produced several separate strands of truth. Reconciliation weaves them back together without dropping a single thread, because every thread arrived already proven.

The Mickai pantheon.

11. Pantheon: anchoring the record to a sovereign Layer 1

Reconciling the records into a shared local history is necessary but not sufficient. A shared history that lives only on systems you control is still a history that could, in principle, be quietly rewritten by whoever controls those systems. For records that must withstand serious scrutiny, you want an anchor no single party can move. That anchor is Pantheon, our sovereign Layer 1, anchored to Bitcoin. When the link returns and reconciliation completes, the sealed record chain is anchored to Pantheon, giving it a settlement point rooted in the most battle-tested proof-of-work security in existence.

> **A record sealed at the edge and anchored to Pantheon is a record that survived the dark and then nailed itself to something nobody can quietly move.**

The reason this matters is the difference between tamper-evident and tamper-evident-and-anchored. The Open Audit Record sealed during the outage is already tamper-evident: alter it and the signature breaks, visibly. But tamper-evidence on a system you control still leaves open the question of whether the whole local history could have been reconstructed wholesale after the fact. Anchoring to a sovereign Layer 1 settled against Bitcoin closes that question. The existence and integrity of the record at a point in time becomes provable against an external chain that no operator, no vendor, and no adversary can rewrite. The dark produced the seal. Pantheon makes the seal unforgeable in time as well as in content.

I call Pantheon sovereign deliberately, because the anchoring surrenders control to no one. The operator does not hand their records to a custodian. They commit a cryptographic anchor, a compact proof, to a Layer 1 that is itself anchored to Bitcoin's settlement security. The records themselves

never need to leave the operator's control to gain that external immutability. This is the principle that runs through the whole architecture: gain the strength of the wider world without surrendering your sovereignty to it. The edge device kept its thinking private through the outage, and now it gains external immutability without giving that privacy up.

This is the closing of the loop that began the instant the cable was pulled. The brain thought alone. The action was sealed alone, post-quantum, on local hardware. The record was queued, durable and ordered, through an outage of unknown length. The link returned. The queue drained, verified and idempotent, into a reconciled history. And that history was anchored to a sovereign Layer 1 settled against Bitcoin, so the work done in the dark is now provable to anyone, forever, without the operator ever having surrendered control of a single byte. That is the full arc, and every stage of it was designed to hold even if the next stage never came.

12. The doctrine: building for the cable that will be pulled

Let me draw the threads into a single doctrine, because the individual mechanisms matter less than the posture that produced them. The posture is this: assume the cable will be pulled, and build a system that does not merely survive that moment but works straight through it, fully and accountably, as though the dark were a normal operating condition. Everything else, the offline-first architecture, the partition-tolerant choices, the local sealing, the durable queue, the reconciliation, the anchoring, follows inevitably from taking that one assumption seriously instead of waving it away.

The industry's default posture is the opposite. It assumes the cable holds, treats disconnection as a rare embarrassment, and ships that assumption to people who do not live in the connected core. I think that is not merely a technical mistake but a quiet abandonment of everyone who operates in the contested, the degraded, and the denied. The places the cloud cannot reach are not a niche. They are clinics after the storm, vessels far from shore, facilities sealed by necessity, operators in jammed spectrum, and a vast quiet majority of the world that does not enjoy uninterrupted bandwidth and never has.

> **Build for the worst link your operator will ever see, seal everything where it happens, and reconcile cleanly when the world returns. The rest is detail.**

If you take one idea from this book, take this. The air gap is not a problem to be tolerated. It is a discipline that makes everything else better. A system built to think alone in the dark is also a system that is private by construction, sovereign by design, and honest about its own limits, because it had to be in order to survive the isolation. The architecture that survives the outage is the same architecture that protects the operator's sovereignty when the link is healthy. Resilience and sovereignty turn out to be the same property viewed from two angles, and you get both from the same set of decisions or you get neither.

I built the SIOS this way because I believe intelligence that needs permission from a distant datacentre to function is not really the operator's intelligence at all. The fifty brains on your own hardware, the

post-quantum seal made the instant an action happens, the durable queue that holds the truth through any length of dark, the clean reconciliation, the anchor to a sovereign Layer 1 settled against Bitcoin: these are not features. They are a single argument, made in engineering rather than in words, that your thinking should remain yours and keep working no matter who pulls the cable or why. The cloud cannot reach the places that matter most. So we built intelligence that does not need it to.

The Mickai pantheon.

APPENDIX · ABOUT THE AUTHOR

Micky Irons

Founder and chief executive of Mickai LTD (Companies House 17166618, registered office 20 Wenlock Road, London, N1 7GU) and named inventor on the Mickai SIOS patent corpus: 101 filed UK patent applications, around 2,234 claims. Trade mark Mickai registered at UK00004373277.

Profiles

mickai.co.uk

crunchbase.com/person/micky-irons

linkedin.com/in/mickyirons

References and further reading

· National Institute of Standards and Technology, FIPS 204: Module-Lattice-Based Digital Signature Standard (ML-DSA), 2024.

· Eric A. Brewer, Towards Robust Distributed Systems, Principles of Distributed Computing keynote (PODC), 2000, the origin of the CAP theorem.

· Seth Gilbert and Nancy Lynch, Brewer's Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services, ACM SIGACT News, 2002.

· Satoshi Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, 2008.

· Pat Helland, Building on Quicksand: Reconciliation-Based Computing and Eventual Consistency, Conference on Innovative Data Systems Research (CIDR), 2009.

· National Institute of Standards and Technology, NIST Special Publication 800-57: Recommendation for Key Management, on the longevity requirements of cryptographic protection.