open nexus

September 26, 2025

The Future Is Open – Rethinking the Operating System

Jenning Schäfer
jenning@open-nexus-os.io

Abstract

For decades, we have built on technologies that were never designed for the world we live in today. x86 and ARM brought us here—but they are closed, inefficient, and shaped by the needs of the past, not the future. We keep squeezing more out of them, but every step forward costs more energy, more complexity, more compromise.

At the same time, something new is growing. RISC-V is open, modular, and free. Rust brings safety and reliability into systems programming. Microkernels promise stability by design. These are not end-of-life technologies—they are at their beginning. And beginnings are where real opportunities lie.

Open Nexus OS is built on this foundation. It is one operating system for all devices: open, modular, secure. Not patched together from yesterday's legacy, but created for tomorrow's needs. Yes, it asks us to let go of the comfortable old. But the price of holding on is higher: poor battery life, fragile security, closed ecosystems, and lost sovereignty.

This is not just a question of technology. It is a question of courage. The courage to stop fixing what is broken and start building what is right. The future will not be shaped by those who play it safe. It will be shaped by those willing to bet on what comes next.

Contents

1 Introduction and Motivation

"When Technology Gets in the Way, It's Not Smart Anymore."

We live surrounded by "smart" devices. Phones, watches, speakers, cars. But too often, the word smart means more complexity, not less. We spend time pairing, syncing, updating, managing settings—when true intelligence would make all of that invisible.

Real intelligence is not about adding more features or pairing it with more powerful Large Language Models. It's about removing friction. It's when your watch unlocks your laptop, your headphones switch seamlessly when a call comes in, or the right controls appear exactly when you need them. That feels effortless. That feels human.

Today, you can experience this—but only inside closed ecosystems. Apple delivers it by locking you in. Microsoft ties it to accounts and policies. Google makes it work by trading your data. The result: walls everywhere. Smooth on the inside, but closed from the outside.

What's missing is an ecosystem without walls. An operating system that anyone can trust, extend, and improve. Something as seamless as Apple, but open by design. Something that does not belong to one company, but to everyone.

That is the promise of Open Nexus OS: One OS. Many devices. Truly open.

—

2 Analysis of Existing Ecosystems

2.1 Apple – Seamless, but in a Cage

Apple shows us how powerful integration can be. Your watch unlocks your Mac. AirPods switch without you touching a button. Everything feels effortless.

But there's a catch. That seamlessness only works because Apple controls every part of the system—and every part of you. The walls of the garden are high, and once you're inside, you can't leave without losing everything that made it work. What looks like freedom is really dependence.

2.2 Microsoft – Integration for Enterprises, Not for People

Microsoft built its empire on the desktop, and it still shows. Windows works best when tied to enterprise accounts, policies, and cloud services. For businesses, it can be powerful.

For individuals? It's fragmented. Mobile integration is weak, often relying on third-party tools. The experience is more about managing devices than living with them. Microsoft makes ecosystems for IT departments, not for everyday life..

2.3 Google & Android – Open in Name, Closed in Practice

Android promised openness. Everyone could build on it. That was the story.

The reality? A patchwork. Each manufacturer building their own version. Updates arriving late—if at all. Interfaces breaking. What should have been an ecosystem turned into fragments stitched together with Google's cloud.

Real integration? You find it almost only at Samsung—because they built their own sub-ecosystem on top of Android. The rest is noise, not harmony.

And at the center, always, is Google. The gatekeeper. The needle's eye everything must pass through. Try to run Android without Google, and you lose half the experience.

Android may be everywhere. On billions of devices. But it is not truly open. It is not truly yours.

2.4 Linux – Brilliant on Servers, Broken in Your Pocket

Linux is the backbone of the internet. Stable. Versatile. Indispensable. But what shines in the server room fails in the devices we carry every day.

The reason is structural: a monolithic kernel. Drivers run in the most privileged mode. One error, and the entire system goes down.

For CPUs and GPUs, good drivers exist. For modems, antennas, power management? Proprietary black boxes. Untouchable, unfixable.

That's why Linux doesn't run on your phone. That's why cars need custom operating systems. Not because talent is missing. But because the architecture was never built for this reality.

3 Why Choose Between Open and Simple?

The lesson is clear: closed systems deliver smooth experiences, but at the cost of freedom. Open systems deliver freedom, but often at the cost of usability. For decades, we've been told we must choose.

We refuse that choice.

Imagine a world where your phone, your laptop, your car, your home all connect—not because one company forces them to, but because the system itself is designed for it. Imagine switching devices as naturally as breathing, with no settings to tweak, no accounts to bind, no permissions to beg for. Imagine technology that feels invisible, because it just works.

That is what Open Nexus OS is built for:

One OS. Many devices. No walls.

No fragmentation. Core functionality built in.

Transparency and security by design.

A foundation for innovation that doesn't ask for permission.

This is not another platform chasing market share. It's a foundation designed to last.

4 RISC-V as the Hardware Foundation

"Legacy Builds Coffins, Not Foundations."

x86 gave us the PC. ARM gave us mobile. Both changed the world. Both belong to the past.

x86 is a skyscraper on sand. Every decade another floor, another patch, another compromise to keep it standing. Powerful, yes. But bloated. Inefficient. An over-engineered relic.

ARM is sleeker, more efficient—but you never own it. It's a rented house. You pay for the keys. You live by someone else's rules.

Neither is a foundation for the next era.

RISC-V is not another chip. It is a clean slate. Modular. Elegant. Free. No licenses. No gatekeepers. No hidden contracts. Anyone can build, anyone can extend, anyone can innovate.

And the beauty: RISC-V is still in its spring. It carries no dead weight of legacy. It grows with today's needs—efficiency, transparency, openness. Fertile ground for an ecosystem that does not yet exist, but must.

Clinging to x86 or ARM means clinging to architectures at the end of their cycle. Betting on RISC-V means betting on possibility. On sovereignty. On the kind of foundation from which futures are built.

5 Microkernel Design: A Foundation That Doesn't Break

"A House of Cards Is Not a Foundation."

Most operating systems today are houses of cards. Every driver—Wi-Fi, modem, antenna, GPU—stacked directly into the kernel. Touch one wrong card, and the whole system collapses.

That's the flaw of the monolithic kernel. It trusts drivers too much. Many are proprietary. Opaque. Untouchable. You don't know what's inside—and when they fail, they don't fail alone. They take the entire system with them.

On servers, Linux shines. CPU drivers are open and rock solid. GPU drivers may be proprietary, but they're predictable enough. That's why Linux owns the data center.

On phones and in cars, it's the opposite. Modems. Antennas. Sensors. Power management. Almost everything is locked behind proprietary drivers. And in a monolithic kernel, all of that runs in the most privileged space. One bug in a closed component, and the system is gone. That's why Linux never conquered mobile. That's why cars ship with fragmented, custom-built systems. Not for lack of genius. But because the foundation itself was wrong.

A microkernel rewrites the rules. The kernel does almost nothing—just scheduling, memory, communication. Drivers and services run in user space, isolated from the core. If a driver crashes, the kernel doesn't. Errors are contained, not contagious.

And when that microkernel is written in Rust, memory safety itself becomes part of the architecture. Whole classes of bugs disappear before they can exist.

That's the difference between a house of cards and a foundation. One collapses. The other endures.

A microkernel changes the rules. It stops pretending every driver deserves blind trust. The kernel handles only the essentials—scheduling, memory, communication. Everything else runs outside, in userspace. If a driver fails, the system doesn't fall. The house stays standing.

This is why Open Nexus OS is built on the Redox microkernel, written in Rust. Rust gives memory safety by design. The microkernel gives fault isolation by design. Together, they build trust not by hope, but by architecture.

We can no longer afford to build on fragile foundations. The future needs an operating system that survives failure. That treats closed drivers as what they are: untrusted guests. That makes resilience the default.

Because an ecosystem is only as strong as its foundation. And this time, the foundation will not break.

6 The Missing Piece

"Without Tools, There Is No Ecosystem."

Every great platform has one thing in common: not just the operating system, but the tools that bring it to life. The Mac had Xcode. iOS had the App Store. Even Windows, for all its flaws, had Visual Studio.

Linux never had that. Not really. Developers stitched together editors, toolchains, package managers. It worked for hobbyists and for servers—but it never created a thriving consumer ecosystem. There was no single place where ideas became apps, where creativity turned into products.

That's the missing piece.

Open Nexus OS will deliver it: a modern, open, and business-friendly development environment. As powerful and polished as Xcode, as flexible as Qt Creator—but licensed under Apache 2.0. Why? Because openness without walls must mean openness in every direction. Not building new cages under the banner of ideology, but creating a space where companies and individuals can innovate together.

Here's the vision: one backend, multiple frontends. Write your logic once, and deliver it everywhere—phone, tablet, desktop, even embedded screens. Yes, it's more work at the beginning. But it's the only way forward if we want real consistency, real simplicity, real scale.

And here's the difference: companies can keep their code closed if they need to. That's not a weakness—it's a strength. Because we don't want to exclude them. We want them in the same boat, helping to shape the standards, investing in the ecosystem, bringing their best ideas.

This is how a new app landscape is born. Not by clinging to old tools, not by enforcing purity tests, but by building the one missing piece that unlocks everything else.

Without it, no open system has ever succeeded. With it, Open Nexus OS can.

7 From Spark to Ecosystem

"Not Features. A Foundation.""

We're not adding gadgets. We're not chasing the next hype cycle. We're building something far more important: the layers of a new foundation.

And like every great foundation, it starts simple, and grows step by step—each layer strong enough to stand on its own, each release real enough to use.

Step 1: Proof of Life Every vision begins with a spark. For us, it's a simple system that boots, lets you log in, and shows a launcher that can switch between desktop and mobile. Not flashy. But it proves the core idea: one OS, many devices.

Step 2: The Core Foundation Next, we add the basics that make an OS real—resource management, libraries, standards. No hacks, no shortcuts. A system you can trust, because it's built right from the start.

Step 3: A New Way to Build Apps Every ecosystem lives or dies with its tools. Our native declarative UI framework makes building apps simple, modern, and consistent. Clean APIs. A design language that feels seamless across devices.

Step 4: Nexus Studio This is the missing piece Linux never had. Our integrated developer studio—think Qt Creator or Xcode, but open and Apache-licensed. One backend, many frontends. Write once, run everywhere. This is where the ecosystem truly begins.

Step 5: Everyday Essentials Media, accounts, notifications, storage, networking. The things users expect, built in the open way.

Step 6: The Invisible Layer Security, updates, package management. The infrastructure no one sees, but everyone relies on.

Step 7: Performance, Tuned We don't ship hype. We ship stability. Measuring, tuning, and optimizing until the system is fast, efficient, and reliable.

Step 8: Ready for Scale Finally, the enterprise features. Device management, compliance, IoT, machine learning. The tools companies need to trust this platform at scale.

And here's how we'll ship it: slice by slice. Not vaporware. Not slides. Every milestone usable. Every release testable. From a booting kernel → to a daily driver → to a thriving ecosystem.

This is not a roadmap of dreams. It's a roadmap of progress.

Each phase builds momentum. Each milestone proves that the vision is real.

8 Community Without Walls

"Openness Only Matters If People Can Join."

Most projects die before they live because they stay locked in labs, hidden behind closed repos.

Open Nexus is different. It's public from day one. Visible on GitHub. Built for real on GitLab. A website that tells the story. A Discord where the first conversations happen.

And it's not talk. The first proof-of-life is already booting. You can see it. It's not polished. It's not finished. But it's real. And real is the only place revolutions start.

9 The Right Person at the Right Time

"Every Movement Begins With Someone Who Starts."

Great ideas mean nothing without execution. Most people wait. I didn't.

I've built software from the ground up. I've led teams across borders. I've managed projects from vision to delivery. And when I saw the cracks in the old world and the outline of the new one, I didn't write a blog post. I wrote code.

That's why Open Nexus exists today—not as a theory, but as a system that runs. Because change doesn't wait for permission. It waits for someone to start.

10 Beyond Funding: Building With Us

"We Don't Want Checks. We Want Co-Builders."

Money is fuel. It speeds things up. But money alone doesn't build the future. People do.

We don't want investors on the sidelines. We want partners in the trenches. Companies, institutions, individuals—anyone who believes that an open ecosystem is the only future worth building.

This is not about buying shares. It's about shaping standards. It's about building something that will outlast us all.

11 A Sustainable Path Forward

"Openness Is Not Charity. It's Strategy."

Free and open doesn't mean fragile. It means scalable. It means unstoppable.

On top of Open Nexus OS, we build Nexus OS: the controlled, certified layer for organizations. Not to wall people in, but to meet the realities of compliance and trust.

A secure fleet of company cars. A watch that holds sensitive documents, like today, but within manageable Guidelines. Tablets in schools. Smart homes with alarms and lights. All powered by the same resilient core.

And the model is clear: enterprise support, hardware certification, cloud services for education and business, an app store that's open by default but trusted where needed. Freedom when you want it. Control when you need it.

This is not idealism. It's a business model. The kind that attracts the best developers. The kind that scales with industry. The kind that wins.

12 Conclusion: The Only Path Forward

"This Is the Moment."

Every era has its turning point.

We had the moment when personal computers left the lab and landed on our desks. We had the moment when the internet escaped the universities and connected the world. We had the moment when the phone stopped being a phone and became the center of our lives.

And now, we are here. At the next moment.

The old foundations are collapsing. x86 is a relic. ARM is owned. Linux never made it to our pockets. Android is "open" but chained to Google. Apple offers perfection—inside a cage.

We can keep patching the cracks. We can keep living in cages. We can keep pretending "open" means freedom while it really means dependence. Or we can stop. Right here. Right now.

Because the future is not written. It is built. And the builders are the ones who dare to walk away from the safe, from the known, from the comfortable.

This is not just my project. It is an invitation. To everyone who sees the cracks. To everyone who feels the weight of walls. To everyone who knows the future doesn't belong to those who wait.

This is your moment. To stop patching the past. To start building the future. To choose courage over compromise.

The future doesn't wait. It gets built. And it starts—with us.