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 – A Masterpiece Without a Manual

Linux is a masterpiece. It runs the internet, powers supercomputers, secures data centers. Stable. Reliable. Indispensable.

But brilliance in the server room doesn't translate to the phone in your pocket.

Because Linux is not an operating system. It's a kernel. The rest is chaos. A thousand distributions, none fully compatible. A developer who wants to publish software has two choices: release the source and hope someone compiles it, or build for a jungle of package formats. And if they want to sell their app? There is no store. No marketplace. No bridge between developers and users.

Freedom becomes fragmentation. Variety becomes walls. What should have been a universal platform turns into a subculture—fascinating for insiders, hostile to everyone else.

And Linux is built for insiders. It asks questions no normal user should face. Architecture, distribution, desktop environment, partitioning, swap space. On macOS, you drag an app into a folder and it runs. On Linux, installation often feels like a computer science degree. An iPhone can go to a child or a senior and just work. Linux cannot.

Even in business, it's always "almost." CPU drivers? Excellent. GPUs? Decent. But fingerprint readers, face login, smartcards? Often broken. Enterprise management, compliance, central control? Missing. And without a company behind it, Linux has no leverage to force hardware vendors to care.

Linux is brilliant. But it is not human. It is a toolbox, not an ecosystem. It has all the freedom in the world—and none of the unity. And that is why, despite its genius, it never became the foundation for our digital future.

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: Architecture That Breathes

"From rigid machines to living systems."

The monolithic kernel is not a foundation. It's a tombstone. It was built because it was easy: one giant block, everything in the most privileged space. Drivers, file systems, networking—one bug, and the whole tower collapses.

But the real flaw runs deeper: Monoliths are the architecture of centralism. One bottleneck, one fragile knot, one point of control. In the 90s, when Linus Torvalds and Andrew Tanenbaum clashed, Torvalds chose the monolith. It was faster. Simpler. Back then, it was pragmatism. Today, it's a prison.

The world has moved on. The desktop under your table has become a planet-wide network. What used to be a process is now a service in a web of systems. In a microkernel, each module is its own server: isolated, replaceable, resilient. A driver crashes? It restarts. A service shifts? The system survives.

The microkernel turns operating systems into organisms: they grow, they shrink, they adapt. And when written in Rust, the last legacy dies—unsafe memory. Security isn't a battle anymore. It's built in.

This is not an academic exercise. It is the correction of a historical mistake. The microkernel isn't just another option—it is the only architectural path that carries us into a modular, connected future. Everything else is a patchwork relic from the age when computers sat alone in dark bedrooms.

6 The Missing Piece – Tools That Build Ecosystems

"Without tools, a platform is just a shell."

Every great platform had one thing in common: it wasn't just the OS, but the tools that brought it to life. Apple had Xcode. Microsoft had Visual Studio. Google had Android Studio. Those weren't accessories—they were the engines that turned platforms into ecosystems.

Linux? It had everything, and somehow nothing. Editors, toolchains, package managers. And yes, it had Qt—powerful, brilliant, the backbone of KDE. But Qt was never the central tool. It was one island in an archipelago. Developers stitched things together, each their own way. What you got was a patchwork quilt—ingenious, but never unified. That's why Linux never became a true consumer ecosystem.

That's the missing piece.

Open Nexus OS will deliver it: not just another editor, but a true studio. A central place where logic is written once, then delivered everywhere—phone, tablet, desktop, embedded screens. Apps that stay lean because libraries live in the system, not inside every package. A user experience that feels consistent, because colors, icons, and controls all come from the same source. Not twenty voices shouting, but one chorus in harmony.

And yes—we will be bold. A single mainline version that dares to deprecate, to break old code when progress demands it. That's how evolution works. That's how ecosystems survive.

The system stays open: developers can still bring in programs built with other tools, from other sources. That's fine. But without one central studio, there is no shared foundation—and no real ecosystem.

Without it, open systems stay fragmented. With it, Open Nexus OS can thrive.

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."

Open doesn't mean fragile. It means scalable. Adaptable. Stronger than any walled garden could ever be.

On top of Open Nexus OS, we build Nexus OS—the controlled, certified layer for organizations. Not to cage people in, but to meet the realities of compliance, trust, and scale. A secure fleet of company laptops. Medical devices with strict rules. Tablets in schools. Smart homes with alarms and lights. All powered by the same resilient core.

And here's the difference: the open base isn't a demo. It's complete. Feature-rich. Fully compatible. We will not repeat the mistake of Android AOSP, where the "open" version sparked fragmentation because everyone had to build their own full system. With Open Nexus OS and Nexus OS, the foundation is the same. One base. Two models.

The business model is simple, and powerful:

An app store that shares revenue with developers, with add-ons and services.

Managed cloud for schools and enterprises.

Certified hardware with licensing and quality standards—so we don't repeat the Windows problem, where cheap machines ruined the software's reputation.

Enterprise services for trust and stability.

And what about adoption? The fear that people won't switch? That companies are conservative, that professionals depend on specialized tools? Steve Jobs faced the same reality with the Mac. His answer: give people the basics. A handful of polished, essential apps—so switching feels safe, not risky. Most people don't need hundreds of programs. Just the right ones.

And younger generations? They're already used to switching—from Windows to macOS, from Android to iOS. Switching is normal. Give them a reason, and they'll switch. Many of them don't even have a Desktop PC. And when enough do, companies follow.

There are two rules we live by:

Build a brand, not just a product. Marketing isn't about features, it's about identity. People join movements, not specs.

Polish before promise. No half-baked MVP. No "trust us, it'll come later." That kills products. Everything we show is ready. Everything works. Like Apple, we ship polished.

That gives us something no one else has: a truly open base, AND our own hardware. Apple would have to build everything themselves. Microsoft is stuck on desktops. Android might follow —but only we can lead.

And here's the bigger play: we're not just competing in existing markets. We're creating a new one. A network of devices that connect by design—not through accounts, not through syncing, but through open standards, built into the core.

That's not idealism. That's a business model. The kind that attracts the best developers. The kind that convinces investors. 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.