www.pipelinepub.com
Volume 22, Issue 10

The Testing Problem Nobody Talks About

By: Anil Kollipara

How AI Is Finally Catching Up to the Complexity of 5G Network Validation

Here's something that doesn't make it into the keynotes at most telecom conferences: testing is broken. Not in the dramatic, everything-is-on-fire sense. More in the slow, grinding way where talented engineers spend weeks chasing problems that shouldn't take weeks. Where the sheer volume of data generated by modern networks has outpaced our ability to make sense of it, and where "do more with less" has become less of a strategy and more of a prayer.

I've spent enough years in this industry to remember when network validation was manageable. You had a defined set of protocols, a handful of vendors, and enough institutional knowledge on your team to troubleshoot most issues over a few cups of coffee. Those days are gone.

The shift to 5G Standalone and cloud-native core architectures has changed the math completely. We're not dealing with incremental complexity. We're dealing with an exponential increase in the number of things that can go wrong, and the number of places you have to look to figure out why.

Think about what a validation team is actually dealing with today. Multi-vendor environments where each component has its own release cadence. Cloud-native infrastructure that's constantly being updated. Thousands of configuration combinations. And when something fails, the evidence is scattered across packet captures, control plane signaling, configuration files, KPI dashboards, trouble tickets, and standards documents. An engineer troubleshooting a single issue might need to correlate data from six or seven different sources before they even have a hypothesis.

A handshake failure between two network functions took over seven weeks to resolve; "the root cause" was determined by an individual (one week after the investigation began) with some help from others along the way. The actual fix only took a few hours. It took longer than expected to find the problem.

Here’s the part nobody sees as a major issue: This is not simply an efficiency problem; it’s a people problem. Engineers able to perform this type of deep-level troubleshooting are scarce. Decades of knowledge about protocols are stored inside these engineers’ heads. When those few individuals spend weeks investigating one particular issue, they aren’t available for the next 100 issues that require their attention. Those new hires will be unable to take up the slack with respect to the vast majority of their tacit knowledge that is required to develop over many years.

Meanwhile, the testing workload keeps growing. Every new 5G feature, every new vendor integration, every software update creates new test scenarios. Teams aren't keeping up. Not because they aren't working hard enough, but because the problem has scaled beyond what human-only workflows can handle.

This is where AI enters the picture, and I want to be careful here because this industry has been burned by overpromising. I'm not talking about dropping a chatbot into a test environment and calling it innovation. The kind of AI that actually helps with network validation looks very different from what most people imagine when they hear the term.

Beginning with the emergence of multi-functional agent-based artificial intelligence systems. Consider a group of unique digital experts, each trained to carry out a specific function. The first expert analyzes raw packet capture data to identify vulnerabilities. The second reviews whether a given network configuration complies with 3GPP standards. The third assesses trends of defect issues based upon previous data. The fourth (coordinating layer), utilizing the results from each of the other three, provides the analyst with a summary report.

The key difference is that the best of these systems does not solely use large language models. While LLMs are useful for natural language interactions, enabling engineers to communicate in a conversational manner versus having to write lengthy query strings, large language models may not be sufficient for the analysis itself. In order to analyze, you require deterministic rule engines and structured domain knowledge. You require systems that know protocol behaviors, not simply text patterns. By using this method, we ground our AI. We reduce the hallucination risks inherent to general-purpose AI tools used in mission-critical areas.

What does this look like in practice? An engineer recognizes that a test has failed. Rather than manually pull packet captures, review configuration files, and page through standard documents, he/she describes the problem in natural language. The AI will ingest the appropriate packet captures and configuration information, correlate its findings to known standard behaviors and historical trends, and deliver a structured root-cause explanation along with prioritized recommendations. As previously stated, my seven-week investigation into this same root cause took approximately two minutes when completed via a similar domain-trained AI. Did the AI "get" smarter than the engineers involved? No. However, it was able to process and correlate vast amounts of structured data in real-time, far exceeding the ability of even the most experienced humans.

It’s worth being precise about what the AI did and didn’t do in that two-minute analysis. The system didn’t make a judgment call about whether the fix was acceptable, whether it introduced risk to other parts of the network, or whether the underlying configuration drift pointed to a process problem that needed to be addressed upstream. Those decisions still belong to the engineer. What the AI did was eliminate the part of the work that doesn’t actually require engineering judgment: pulling the right packet captures, lining them up against the relevant standards, flagging the anomaly, and presenting the evidence in a form a human can act on. In the old workflow, that data correlation work consumed most of the seven weeks. The judgment call took an afternoon. When you compress the first part to minutes, you don’t replace the engineer. You free that engineer up for the part of the job that actually

needs them, and for the queue of other issues that have been sitting on hold waiting for someone with the right expertise to look at them.

Beyond speeding up troubleshooting, these same AI capabilities also support accelerating test development. Engineers can define what they wish to test via natural language and receive automated test case generation that reflects true protocol behavior and standards requirements. They can also modify existing test scenarios through guided prompts, thereby eliminating manual script edits. This will not eliminate the role of engineering judgment, but will amplify it.

Regarding organizations that maintain secure environments —and this would include most service providers and infrastructure teams— while the capability described above is important, so is the deployment model. Most viable implementations operate exclusively on premises, entirely within the client's secured environment. All data, all processing, all proprietary knowledge remains under client control. No test data nor KPI's ever leave the facility. To those teams whose primary responsibility involves maintaining their clients' environments under very strict security guidelines, this is less about being desirable and more about being mandatory.

This isn’t a theoretical concern. The data flowing through a telecom validation environment includes a lot more than test traffic. Packet captures from a live or pre-production network can contain signaling that reveals subscriber identifiers, routing topology, security parameters, and roaming relationships. Configuration files describe how the network is built. KPI sets describe how it’s performing, and where it’s vulnerable. None of that should be sitting in a third-party cloud, even one with strong security posture, and in many jurisdictions it legally cannot. Regulators in Europe, the Middle East, and parts of Asia have tightened sovereignty rules considerably, and operators with government or defense customers face additional restrictions on top of that. Any AI capability that requires shipping data outside the operator’s perimeter is a non-starter for a meaningful share of the global market. The teams I talk with don’t want a vendor demo of cloud-hosted AI. They want to see the same capability running inside their lab, behind their firewall, with their data, before they’ll take it seriously. That’s the right instinct, and it’s shaping how this category is going to develop.

I think we're at a genuine inflection point. Not the kind the marketing slides talk about, but a practical one. The complexity of 5G networks has created a validation bottleneck that traditional approaches simply can't solve at scale. AI-driven testing isn't a futuristic concept anymore. It's becoming a necessary capability for any organization that wants to keep pace with network evolution.

And it's worth thinking about what comes next. As networks move toward 6G architectures, which will be even more distributed, more AI-native, and more complex, the testing challenge will only intensify. The organizations that build AI-assisted validation into their workflows now won't just be more efficient today. They'll be positioned to handle the next wave of complexity without having to reinvent their approach from scratch.

The engineers doing this work deserve better tools. Not tools that try to replace their expertise, but tools that handle the grunt work of data correlation and pattern matching so they can focus on the problems that actually require human judgment. That's not hype. That's just good engineering.