ESSAY

Building R&D on Soft Ground

What structural engineers know about building serious work where the infrastructure is missing

Chima Emmanuel Akuegbo

The Leaning Tower of Pisa began to tilt before it was finished. Its builders laid a shallow foundation on soft clay and sand, and by the time they reached the third storey the ground on one side had started to give. Then, for reasons nobody fully knows, the work stopped for close to a century. That pause, by most engineering accounts, is a large part of why the tower is still standing: the clay had time to consolidate under the weight it already carried before anyone asked it to carry more. The tower is a monument to bad ground, and also to the fact that bad ground can be built on, if you respect it.

I think about that tower a lot, because I build research systems in Nigeria, and the ground here is soft.

That is not a complaint; it is a site report. The infrastructure a well-funded lab never thinks about (usable data, local compute, trustworthy measurement, steady electricity) is missing or unreliable here. That does not make serious work impossible. It changes the method. And the discipline that has thought hardest about building safely on ground you did not choose, with materials you cannot fully trust, under loads you cannot fully predict, is not computer science. It is structural engineering.

I build models and pipelines, not bridges. But I have come to believe that the habits of a good structural engineer are exactly the habits that keep a small team alive when the infrastructure underneath it is missing.

The ground we actually build on

Before a foundation is designed, the site is investigated. Here is ours.

Data you cannot legally use. People say research here is short of data. That is only half true. Large datasets exist, but their licences often forbid the use a local builder most needs: non-commercial only, no generative use, share-alike terms that spread to everything built on top, caps on how many people you may serve. In my own work the binding constraint on data was legal before it was technical. A dataset can be big, relevant and free to download, and still be ground you are not allowed to build on.

Compute you rent from abroad, on credit that expires. There are no accelerators

in my building, or my city. Compute is rented by the minute in another country, in foreign currency, often from prepaid credit with an expiry date; researchers who map AI compute describe much of the world, Nigeria included, as a compute desert. One entire multi-phase programme of mine ran on about US$140 of rented time. That works only if no run is wasted, and if you remember that when a rented machine stops, its disk is wiped and anything not exported is gone.

No trusted way to know if you are right. The least visible deficit, and probably the most damaging. There are few calibrated automatic judges, no standing panels of paid expert reviewers, and few open reference sets. Some instruments that do exist are quietly wrong for local conditions, by margins larger than the improvements you are trying to measure.

Power that comes and goes. Nigerian firms in a 2014 World Bank enterprise survey reported more than thirty outages in a typical month, and the African Development Bank's 2026 African Economic Outlook puts the share of firms that own or share generators at around seven in ten. The national grid collapsed repeatedly in 2024: eight times by the Minister of Power's count, more by press tallies. The workstation, the instrument and the network link can all fail mid-session, so long jobs must run unattended and checkpoint often enough that an outage costs minutes, not days.

None of this can be fixed by one team. It is infrastructure: shared, expensive and public in character. So a method for working here has to treat it the way an engineer treats soft clay or high wind: not as an excuse, but as a fixed condition of the site that the design must carry.

Here is the method, in eight principles.

1 Investigate the site before you design

A geotechnical engineer does not choose a foundation and then look at the soil. The investigation comes first, and what it finds can change everything: piles instead of a raft, a different footprint, sometimes a different site.

In research, that means characterising every source before you design around it: what its licence actually says (and whether two of its own pages disagree), how big it really is, how it was collected, who it represents, and a few measured properties. Then design from what you found, not from what you hoped.

On one project, the audit excluded the two largest datasets available to us, although either would have improved our headline numbers. It also found that most of what remained came from one narrow kind of source, in one careful, formal style. That predicted, months before any model existed, that whatever we trained would inherit the narrowness. It did, but we had seen it coming and were already re-weighting sources and measuring against real references. The same audit found many automatic labels from off-the-shelf tools implausible. We did not use them.

The cross-field version is easy to see. A health team whose patient records were consented for care, but not for research, has a site problem, not a modelling problem. A crop trial on a plot whose history nobody recorded is building on unsurveyed ground. In each case, the cheapest moment to find out is before the first pour.

2 Every load needs a path to the ground

Structural engineers talk about the *load path*: the continuous chain of members and connections that carries every force down to the foundation. A beautiful upper storey on a broken path is not a strong building. It is a building waiting for the load to discover the break.

The starkest illustration is the Hyatt Regency walkway collapse in Kansas City in 1981. A late change to how suspended walkways hung from their rods, from one continuous rod to two offset ones, quietly doubled the load on a single connection. The load path had been broken at one joint, and 114 people died.

In research, the load path is provenance. Every claim you make and every artefact you release should trace an unbroken chain back to foundations you are allowed to build on, through every intermediate model, dataset and tool. Constraints travel down that chain. If one source anywhere beneath your release is non-commercial, the whole structure is non-commercial. Share-alike terms behave like load: an obligation on the foundation becomes an obligation on everything above it.

We took this literally. Because some permitted sources were share-alike, we released trained models under a share-alike licence and code under a permissive one, the conservative reading of an unsettled legal question. A useful non-commercial component was allowed to act only as an evaluation judge, off the load path of anything we shipped. Outputs of experiments that touched restricted material never entered a release.

The same logic governs a clinical model built on consented records, a seed variety whose rights trace back to someone else's germplasm, and a materials sample whose chain of custody must be unbroken before anyone believes the measurement. A breach discovered late does not damage one floor. It invalidates everything above it.

3 Margin is the design

Structural codes do not ask whether a beam is strong enough. They ask whether it is strong enough *after* the load has been multiplied by one factor and the material's strength divided by another. Those safety factors are where the honesty lives: they say, in numbers, how much we do not know. Engineers also separate *ultimate* limit states (collapse, injury) from *serviceability* ones (a floor that bounces, a door that sticks), which need different evidence and different margins.

Research claims deserve the same treatment. Before calling a difference an improvement, I ask: how much does the result move if I simply run it again, how biased is the instrument, and how strong is the claim?

In one programme, re-running the identical experiment moved our main error measure by a small but real amount. So differences of that size between old and new systems were reported as ties, even when they favoured the new one, which is to say even when I wanted to believe them. A selection trick that picked the best of several outputs using the same metric it was later scored on cut errors by about a third; we reported that as an upper bound, because the judge had helped choose the winner. And a system that clearly cleared the ultimate bar (it worked, reliably, on every case we cared about) but had no human verdict on quality of use was released as *experimental*. Serviceability was

exactly where the infrastructure gap showed, and the release said so.

A yield claim from a single season has no safety factor. A mini-grid serving a clinic and a row of shops should treat the clinic as the ultimate limit state and the shops' comfort loads as the serviceability one.

4 Buy the beam, build the joint

Much modern construction is prefabricated: beams, panels and trusses are made and certified in a factory, where conditions are controlled, and joined on site, where they are not. The site crew builds connections, not everything.

When compute is rented and data is scarce, this is the most useful idea I know. Large pretrained components, whose capacity was paid for by someone else's infrastructure, are prefabricated members. Freeze them, build only light task-specific parts on top, and treat training a large component from scratch as a decision that needs its own justification.

We once ran the comparison directly, on the same small budget and data. A model built from scratch was still useless after an hour; a fine-tune of an existing pretrained model was doing the job within about thirty-five minutes. Everything we built after that was a fine-tune. In another experiment, two frozen components joined by a pair of tiny trained connectors took minutes to train and cost about US$1.50, and the result went from near-useless to genuinely good on the task we cared about. Unfreezing more of the components added nothing.

But certification has limits. A beam rated for one load case may have little capacity for another, so test the member under *your* load. The obvious off-the-shelf choice for that experiment was tested first and turned out to be weak on exactly the cases we needed. A health team adopting an imaging model validated on foreign patients is in the same position: certified, just not for this building, until validation on local patients says otherwise. Standard inverter modules, released crop varieties and commercial chemical precursors are all prefabrication, and all need the same local check.

5 Pour in stages, and stop at the hold points

Concrete is poured in stages, and a good site runs on an inspection and test plan. Some checkpoints are *witness points*, where an inspector is invited to look. Others are *hold points*, where work stops until the inspection is passed. A defect found at a hold point costs an afternoon. The same defect found after the next pour costs the floor.

Under expiring credit, this is not bureaucracy. It is survival. I now divide every programme into stages, each ending with a test whose pass criteria are written down before the stage begins, and the next stage is funded only when the gate is passed. Cheap tests that discriminate well go first. And the gate has to sit early enough that a failed stage still leaves budget to build a corrected one.

The first stage of one programme was a pair of smoke tests costing under US$10 that chose the architecture for everything after. Each later phase ended with a written report and a stop, with machines shut down and results exported before the next was approved.

The most important hold points are human. A quick fix once improved two of our automatic metrics, and a person who actually used the output judged the untreated

version far better. The fix was withdrawn and the problem went back to the stage where it really lived, where it was solved properly. Another time, one sharp comment from a reviewer exposed a defect none of our checks measured, and it became a new check.

This is what Pisa's long pause did by accident: it let the ground answer before the next load went on. Pilot cohorts before scale-up, on-station trials before on-farm trials, a pilot mini-grid before a roll-out, bench before plant: the same discipline.

6 Test the real material, and test the testing machine

Engineers accept concrete and steel on tests of the material actually delivered to the site, not on a brochure or a similar batch from elsewhere, and the testing machines are themselves calibrated against references. This principle caught two of my most expensive near-misses.

The first was a shortcut that looked great on our own tests and moved nothing in the real world. The real data one part of a system needed did not exist under a usable licence, so we generated synthetic data to stand in and ran a controlled comparison. On a test set built the same synthetic way, errors fell by roughly two-thirds. On real held-out data, the improvement was a rounding error, well inside the noise of simply re-running the experiment. The synthetic material had passed a test made of itself. The real-data model stayed the default.

The second was the testing machine. When we finally pointed a popular automatic judge at real reference examples, the ground truth we were trying to approach, it scored them *lower* than every system it was meant to judge. It rewarded a polished artificiality, and it would have steered us steadily away from our target. We demoted it to a minor sanity check and built simple measures anchored on real references instead.

So: admit an intervention only if it improves the real target, not a proxy, and calibrate every judge on real references before it chooses anything. Satellite crop estimates calibrated on another country's fields, energy forecasts built from borrowed demand profiles, and a lab instrument never checked against a certified reference sample are all uncalibrated machines until proven otherwise.

Load testing belongs here too. London's Millennium Bridge closed within days of opening in 2000, because crowds walking in step made it sway sideways; it reopened two years later with dampers. Test under the loads you will really meet: the slowest one-in-ten requests rather than the average, interruptions, strange inputs, a clinic's device through a real outage in real heat. In one pipeline of ours the average looked fine and the slow tail did not; one stage dominated it, and fixing that stage did more than tuning all the others.

7 Give every load a second way down, and let the parts move

Good structures are redundant: if one member fails, the load finds another path instead of bringing the structure down. They also have movement joints, deliberate gaps that let one section expand or settle without cracking its neighbour. Both accept that parts will misbehave, and keep the misbehaviour local.

For any component working at the edge of its competence, I want three things: a

detector that recognises failure, a retry or alternative, and a final safe state that is always acceptable even if less capable. One weak link dominates the reliability of a chain, and its confident wrong answers get amplified downstream. So our weak components sit inside validators that check output, regenerate under stricter rules, and fall back to something simple and safe. When we fine-tuned a better version of one component, the original stayed on as the fallback. The detector matters as much as the fallback: a failure nobody notices never reaches the safety net.

Movement joints are about coupling. Where the data to train one large all-in-one system did not exist, we built a chain of smaller parts joined by simple, documented interfaces, so each could be improved, replaced and released on its own. Code and models carried different licences in separate files. Old and new versions shipped side by side behind a switch, so a regression in one area forced nobody onto the new version.

A clinician behind an automated triage step, mixed cropping against one crop's failure, a battery path when the grid drops, separate generation, storage and billing layers in a mini-grid: the same two ideas elsewhere.

8 Read the wreckage

Henry Petroski spent a career arguing that engineers learn more from failure than from success. The Quebec Bridge fell during construction in 1907 partly because the bridge's own weight had been underestimated and not rechecked when the span grew. The Tacoma Narrows Bridge tore itself apart in moderate wind in 1940, and the study of why reshaped how engineers think about wind on long spans. Each time, the profession found the mechanism and wrote it into the next generation of practice.

In a small team this habit pays back most. Treat every anomaly as a specimen: reproduce it in isolation, fix the mechanism rather than the symptom, add a check so it cannot return silently, and record negative results as carefully as positive ones.

Our worst failures were silent. One model trained for a whole run and produced rubbish; a learning signal that should have moved stayed flat throughout, and nothing alarmed. A small isolated test found a single setting wrong by orders of magnitude, and a short re-run showed the signal falling at once. In another case, a rule designed to stop a system getting stuck in repetitive loops was also, quietly, forbidding a perfectly legitimate behaviour that happens to look like repetition. Relaxing the rule for that case fixed it, with a cap added after an uncapped version produced exactly the runaway the rule existed to prevent.

Repairs fail too. One fix that solved its target problem doubled the rate of a different defect; a later stage built to penalise that defect brought it down but caused a smaller regression elsewhere, which we disclosed rather than buried. Knowing what did not work is often the most expensive knowledge a small team owns. Crop-failure post-mortems, outage root-cause reviews, misdiagnosis reviews and failure analysis of rejected batches are where other fields keep theirs.

What the country should build

Everything above lets a small team build soundly on poor ground. But it is a way of coping with missing infrastructure, not a substitute for it. Safety factors are expensive. Every margin added for an uncalibrated instrument, every stage gated because credit expires, every dataset we cannot use, is capacity a better-served team spends on the work itself. Each deficit has a direct remedy.

License publicly funded data for use. Data collected with public or donor money should carry licences that permit commercial and generative use with attribution, stated consistently wherever it is published. Restrictive terms on public-good data push local builders onto smaller, narrower foundations.

Provide compute on predictable terms. Shared national or regional accelerator access, or credit without short expiry dates, would let teams plan staged work instead of racing a deadline set by a billing system.

Build an evaluation commons. Paid, standing panels of domain experts; open, held-out reference sets for local clinical tasks, crops, energy profiles and materials; published calibration data for the automatic judges people already use. Nobody asks every construction firm to build its own materials lab and certify its own steel.

Make research facilities power-resilient. Reliable power at universities and research hubs is a precondition for laboratory science and for any local compute at all.

Notice where these sit. Not in the members, the models, devices and clever ideas Nigerian teams are already building, but in the foundations and instruments: the parts no single team can pour by itself, and that everything else rests on.

> **THE SHORT VERSION**
>
> - **Investigate the site:** audit licences, consent and the real shape of the data first.
> - **Keep the load path continuous:** every claim traces back to ground you may build on.
> - **Design with margin:** repeat-run noise, instrument bias, claim strength.
> - **Prefabricate:** freeze what others certified, build the joints, test under your load.
> - **Stage and hold:** cheap, decisive gates early; humans as hold points.
> - **Test the real material:** proxy wins are not wins; calibrate every judge.
> - **Build redundancy and joints:** detect failure, fall back safely, decouple.
> - **Read the wreckage:** find mechanisms, add checks, record what failed.

The tower in Pisa still leans. It was never straightened, only stabilised, by people who measured the ground carefully and removed a little soil from under the high side. That is the honest ambition for building here, for now: not to pretend the ground is firm, but to know it so well that what we build on it stands anyway. Build for the ground you have, with margin for the ground you cannot see, and keep asking the country to pour a better foundation.

Notes on sources

The experiments described are my own. Pisa: J. B. Burland and colleagues on the tower's history and stabilisation (Ingenia; *Soils and Foundations*, 2003). Hyatt Regency: US National Bureau of Standards investigation, 1982. Quebec Bridge: Royal Commission report, 1908. Millennium Bridge: Arup's published account and BBC reporting, 2000 and 2002. Tacoma Narrows: Washington State DOT history. Power: World Bank Enterprise Survey, Nigeria 2014; AfDB African Economic Outlook 2026; Minister of Power's statement to the Senate, January 2025. Compute deserts: Lehdonvirta, Wú and Hawkins, Oxford Internet Institute (AIES 2024). Henry Petroski, *To Engineer Is Human* (1985).