easl

WHITEPAPER

Lessons in Data Movement from the Field

What one founder’s hard-won experience reveals about the hidden costs of patchwork systems and the case for adaptability.

BY ANDY GUY, HEAD OF PRODUCT, EASL

When Growth Outpaces the Plumbing

As co-founder and CTO of a high-growth, tech-enabled insurance lead-gen company, I didn’t know what I didn’t know. Torchlight scaled fast, generating hundreds of thousands of new records every day, each one critical to revenue. But while we were celebrating growth, the infrastructure underneath us was reaching its breaking point. The lesson I learned: Growth accelerates everything, including the cracks in your architecture.

What failed us wasn’t the volume of data, but the lack of adaptability. Our systems couldn’t evolve at the pace of the business, and the burden fell back on people. The result was wasted time, exhausted teams, and trust in the data that steadily eroded.

This paper unpacks that experience—not as a cautionary tale, but as lessons from the field, and as a look at how adaptive data movement changes the game.

The Marketplace That Moved Too Fast for Its Own Systems

I’ve spent my career building data-driven businesses. I’ve seen firsthand how fast growth can create hidden problems, and how quickly those problems can spread if the underlying systems aren’t built to adapt.

Torchlight Technology Group was a digital marketing and insurance lead-generation company I co-founded. We built a real-time marketplace for Medicare and health insurance leads, connecting consumers with call centers and insurers in the moments they were shopping. At first, the business scaled beautifully. But beneath the surface, our data infrastructure was falling behind.

We had the customers, we had the traffic, we had the capital. What we didn’t have was a system designed to move and adapt with the pace of growth.

That’s the hinge in this story: Growth was outpacing the plumbing beneath it. And once that happens, it’s only a matter of time before cracks turn into fractures.

The Torchlight Challenge

Torchlight positioned itself as more than a lead-gen shop. Our core focus was building and using proprietary technology that connected clients to the information they needed quickly, reliably, and at scale.

We took a multichannel approach—buying our own media, generating leads from in-house properties, and partnering with some of the most respected media players in the industry. Every click and every form fill represented a consumer looking for guidance on health insurance or Medicare.

Behind the scenes, that meant handling 150,000 transactions a day, each with over 30 data points. That's more than 4.5 million individual fields flowing through the system daily. And every single record had to be validated, processed, and matched in real time.

Speed mattered. If a record stalled, the consumer's attention moved on and the lead lost value. Accuracy mattered. If a field went missing or a schema shifted, buyers noticed and they complained. Our entire business depended on getting data movement right.

> The challenge was technical, but it was also existential. If we couldn't move data reliably, we couldn't sell leads. And if we couldn't sell leads, we didn't have a business.

The Homegrown Approach

We did what most fast-growing companies do: We built it ourselves. Torchlight's philosophy was "AWS-first, everything in-house." We believed that by building internally, we'd control our own destiny and create institutional knowledge that no vendor could match.

Our architecture reflected that mindset:

- **PHP scripts** to batch data and run overnight transfers
- **AWS Data Pipeline, Kinesis and Firehose** to move rows of data
- **Redshift** as the warehouse of record

On paper, it looked modern. In reality, it required far more maintenance than we'd expected.

The hidden costs piled up quickly:

- **The wrong skill sets:** We had software developers moonlighting as data engineers, which meant high salaries applied to the wrong kind of work. Data engineers could have handled it more effectively and at lower cost, but the role was still emerging at the time, so we leaned on software developers instead.
- **Friday night deployments:** Schema changes or updates meant late nights, with someone babysitting scripts and hoping nothing broke.
- **Fragmented solutions:** As AWS released new generations of data movement tools, we adopted them incrementally. While each successive tool offered added value, none delivered a comprehensive solution, and each came at a higher cost. The result was a stack of data pipelines being executed across disparate systems all within AWS. This fragmented solution never fully cohered.
- **Pipeline fragility:** A new field in the transactional database rippled through every layer, forcing updates across scripts, pipelines, and warehouse tables.

We thought we were building a modern, cloud-first system. What we really built was a delicate machine that demanded constant human attention.

Where Things Went Wrong

The cracks showed up in ways that any executive would recognize:

Data quality issues: Reports always came with caveats. We'd tell investors, “This column isn't quite right, just ignore it.” At first, people tolerated it. But after the third or fourth time, credibility eroded. “Mostly right” quickly became indistinguishable from “wrong.”

Audit trail headaches: When Torchlight was acquired, the new parent company's CTO drilled into our reporting. Month after month, he asked why certain numbers didn't reconcile. Each time, we had long explanations about schema changes, pipeline errors, or timing mismatches. Instead of building trust, the caveats gradually chipped away at it.

Operational overhead: Our smartest people spent their days monitoring dashboards, fixing broken scripts, and explaining inconsistencies. Even I, as CTO, ended up maintaining legacy scripts. Every hour spent firefighting was an hour not spent innovating.

Replay and recovery gaps: If a pipeline failed, we couldn't easily replay the data. AWS tools weren't designed for rewinding; once a row was gone, it was gone. That meant lost records, incomplete reports, and long nights trying to piece together what happened.

The cumulative effect took its toll. Growth that should have been a flywheel turned into a drag. Talent that should have been building the future was stuck maintaining the past. And credibility that should have been earned with data was undermined by its inconsistency.

What Adaptive Data Movement Would Have Changed

In hindsight, the takeaway isn't limited to pipelines or monitoring. It comes down to the philosophy behind how systems are built. At Torchlight, we built systems in silos—scripts over here, warehouses over there, developers stretched across all of it. Change was disruptive because adaptability wasn't built into the operating model.

That's exactly what DataDevOps solves. It treats data infrastructure the way modern teams treat software: versioned, automated, observable, and resilient by design. In that model:

Change is expected, not feared
Schema updates or new sources are absorbed naturally instead of cascading into failure.

Quality is constant
Monitoring, validation, and lineage are embedded in the workflow, not bolted on afterward.

Recovery is routine
Pipelines can rewind, replay, and self-heal, eliminating the sleepless nights we faced.

Teams focus on outcomes
Developers build products, engineers optimize flows, and the system adapts around them.

Back then, DataDevOps wasn't part of the vocabulary. The tools and practices we take for granted today simply weren't available. If they had been, growth would have been fuel instead of friction. The system could have flexed as the business expanded, trust in the data would have scaled alongside it, and our team's energy would have been put to better use.

Lessons for Today's Leaders

Torchlight's experience isn't unique. Many companies—maybe most—are on the same path. The manifestations vary across organizations, but inevitably, ad hoc systems break down as demands intensify.

Here are the lessons I'd offer to any leader staring down the same challenges:

Recognize the inflection point
Every system works until it doesn't. The trap is assuming that because it worked yesterday, it will work tomorrow. The moment you hear caveats or exceptions in reporting, the moment your best engineers are doing damage control instead of building, you've hit the inflection point. Act before the cracks widen.

Align skills with the issue at hand
We leaned on software developers to build and maintain data pipelines because it felt like the closest match. But it was the wrong skillset at the wrong price. A developer can write scripts, but it takes a data engineer to design pipelines that scale. The best organizations hire and deploy the right expertise from the start, saving money and avoiding burnout.

See data movement as leverage
Too many leaders treat it as plumbing—important but invisible. The truth is, it's the operating system of the business. Pipelines dictate how fast information reaches decision-makers, how trustworthy reports look to investors, and how smoothly new products get to market. When data moves cleanly, everything else accelerates almost invisibly.

Weigh build vs. buy honestly
At Torchlight, our "AWS-first, everything in-house" philosophy looked smart on day one. Over time, it turned into an expensive, fragmented system that was harder to hold together than to scale. Custom builds always come with hidden costs, including developer hours, lost opportunities, and endless upkeep. The strongest teams know when to invest in building and when to lean on proven tools that already work.

Future-proof the foundation
Stability is temporary. Change is constant. Yet too many systems are designed for the illusion of stability rather than the inevitability of change. Torchlight's architecture punished every schema update and new field we introduced. Future-proofing involves building systems that absorb change with minimal disruption, so growth feels like momentum instead of a breaking point.

The Window Is Closing

The data infrastructure decisions you make today will determine whether your company scales smoothly or hits a wall in 18 months. While your competitors wrestle with the same pipeline problems Torchlight faced, there’s an opportunity to leapfrog them entirely.

Speed matters more than ever. Markets reward companies that can iterate quickly and respond to change without breaking their systems.

Success belongs to leaders who see the shift coming and act before a crisis forces their hand. Some have already acted. Others are still debating whether the problem is real.

Your move.

easl

Built on Experience, Designed for Tomorrow

EASL was conceived by a team with 35+ years of experience moving data at massive scale. Our platform integrates this deep expertise with cutting-edge technologies to solve the acute challenges of scaling data implementation and processing capabilities that face any high-growth company. Our SOC2 Type I & II certified platform operates with zero-record-loss according to the highest compliance, audit and security standards.

Learn how EASL can help you get your data right.

Visit: easltech.com

ABOUT ANDY GUY

Andy Guy is a technology leader, entrepreneur, and investor with more than 15 years of experience building and scaling data-driven businesses. He co-founded Torchlight Technology Group, growing it from a three-person startup into a 40+ employee company and leading its successful acquisition. At Torchlight, Andy served as CTO, architecting the company's proprietary platforms for digital marketing and insurance lead generation. He has since held technology leadership roles with Intellivets and SunshineMD, and today is EASL's Head of Product.

Email: aguy@easltech.com