Breaking the monolith

A CIO's guide to avoiding legacy IT

/thoughtworks

Break the cycle.
Break the monolith.

Breaking the monoliths of legacy IT is a never-ending challenge for IT leaders. Complex interdependencies make legacy systems almost impossible to sunset. Ever-changing demands see productive systems become unnecessary bloat almost overnight. And even today's shiniest new capabilities seem all but destined to become tomorrow's legacy estate.

It's a problem that virtually everyone in the industry acknowledges. There are countless blogs, webcasts and think pieces exploring the benefits of legacy modernization and telling you why rooting out legacy IT is such a valuable endeavor. Yet, very few people seem interested in understanding where legacy IT really comes from, and how to prevent it before it happens.

From what we've seen, that largely seems to be down to one major misconception: that legacy IT is unavoidable and occurs simply when current technology isn't fit for purpose anymore.

The reality however is far more nuanced. There are a huge number of factors that contribute to the build-up and complexity of legacy IT. And contrary to what some still believe, many of them are completely avoidable.

We've created this guide to equip CIOs with technology strategies that would align with their future business goals. By building a technology backbone that is resilient, competitive, and scalable – even in the face of uncertainty and volatile market conditions. Ultimately ensuring technology acts as an enabler to reducing time to market, spurring innovation with

customer-centric products and services, and inducing agility across the organization.

Inside, we'll explore how attitudes to digital change and cultural constraints, implementation and maintenance factors all influence the build-up of legacy IT — challenging the way things are done today and offering practical advice on how these processes can be improved.

We hope it helps you in your journey towards modernizing your technology estate and building an agile future organization, which is fast and responsive to customer needs.

Isa Goksu

Director of Technology, Financial Services,
Thoughtworks UK

The first step:

Shattering the illusion of stability

The first step: Shattering the illusion of stability

Despite the disruption to the global economy in 2020, something digital natives are continuing to see great success — even during unforeseen circumstances like COVID-19. Amazon, Apple, Zoom, Netflix, Babylon Health, and Ocado all added roughly 30% to their share prices or net profits in the first half of the year.

One could argue this is due to lockdowns, increased health concerns and there simply not being much to do other than order groceries online and watch TV shows at home. However, consider these questions: Would Netflix be as successful if it had remained a DVD rental company? And would Babylon be as successful today if it forced patients to come to its practices physically?

Success or failure has nothing to do with the size or age of the organization. Being successful today has far more to do with business agility and the ability to adopt relevant technologies within your "window of stability"— or drive change when you're under no pressure to do so.

Stability is something virtually every leader values massively. That raises a few important questions:

- Why is it that we have this idea of stability being so important and we keep introducing control and governance mechanisms to make change extremely difficult?
- Why is it that all digital-native businesses — like Amazon — look very stable, yet still respond to change incredibly quickly?

- What is it that we want that we blindly believe stability will give us?

Before you answer those, first consider that, at the most fundamental level, stability is simply an illusion. Nothing is stable in the universe, yet everything looks stable. From every single particle of the universe to the giant systems, they all have constant vibrations, work, effort, or energy flowing through them — which from a distance gives us the illusion of stability.

Thinking about that in an enterprise context, what happens in reality is that there's an enormous amount of work being put in behind the scenes to make 'stable' enterprises look stable. There's a constant stream of tactical decisions being made, small changes being absorbed and failed projects being acknowledged as a cost of doing business. It's a constant low-level change that, when observed from the outside, looks like perfect stability.

So, if your aim is to improve stability, the real question you should be asking is, "How can I make the change an organic part of my organization?"

The challenge of enabling organic change in a complex enterprise

The challenge of enabling organic change in a complex enterprise

Change in any enterprise is extremely hard to manage, simply because there are so many moving parts to consider and align. In the end, an enterprise is a complex adaptive system — a system in which a perfect understanding of the individual parts doesn't automatically convey a perfect understanding of the whole system's behavior. As an example, think of a flock of birds.

That's exactly why we all have different experiences interacting with these systems in real life — from service quality we receive as customers, all the way down to being an employee in one of these organizations. It's also the reason why so much change is unsuccessful in these systems — people or teams devise solutions that don't fully grasp or solve the problems of the organization as a whole.

Once you acknowledge that, it's all too easy to accept it as a given and concede that large organizations will always by their nature be complex adaptive systems. However, we believe it's far more valuable to ask, "How did things get to be this way?"

Enterprises never become complex by design. They get that way over time, as controls, processes, systems, technology and people become more numerous and dispersed. Decisions get made by more people, all with slightly different goals and eventually you end up with a complex structure where everyone simply tries to keep things stable and maximize stability.

Whether you're looking at things from a core IT system or cultural standpoint, that is 'the monolith'. It's not a single,

hyper-complex system that someone decided to design and deploy one day — it's a complex web of accidental complexities and unintentional consequences.

Each one of the decisions made by an organization was made for a reason. Generally speaking, they'll have been made by intelligent teams and leaders with good intentions. Yet still, together, these decisions gradually conspire to restrict the organization and limit its agility. Small decisions and deployments cluster together to form islands, then those islands merge to become continents — and continents are extremely hard to move.

Once those continents have formed and everyone starts believing that they're unbreakable, the aim of the enterprise simply becomes keeping things stable — because people just can't visualize a better way of operating that doesn't involve that monolith in some capacity.

How ‘legacy infrastructure’ is born

How ‘legacy infrastructure’ is born

Now, let’s walk through a journey towards the end-state of a legacy system. Within this journey, we’ll have the chance to look in detail at the big bone structures of IT anatomy and see how accidental complexity manifests in a core IT setting.

The two underlying causes of this accidental complexity are:

Build-time compromises

Insufficient care & maintenance

Defining ‘accidental complexity’

When we refer to ‘accidental complexity’ in this guide, we’re talking about the myriad ways that new architectural design decisions or processes can add complexity to the existing estate (or within itself) without meaning to.

Nobody designs IT to be intentionally complex, so practically, almost all kinds of complexity as we know them fall under this umbrella. The term accidental complexity is a reminder of that — that most complexity can be avoided through better planning, documentation and management and stronger processes.

The impact of build-time compromises

Our journey starts when there's a desire to make a change to the business. In most cases, there's a business leader driving the particular change and there's also a clear aim and business value associated with it. Enterprise leadership has blessed the program with a deadline, initial budgetary approvals have been granted and clear directions on how to spend the associated ballpark figure have been laid out.

At this point, the change delivery team is set up and they're given a high-level understanding of what needs to be done. The process might vary depending on individual enterprise governance and control mechanisms but for the most part, this is how all change journeys begin.

This is also where compromises start occurring and accumulating.

It's absolutely crucial to be aware of these compromises if you want to prevent them from happening. The good news is, once you understand how, where and when they happen, they're all relatively easy to address — ensuring that your next big change doesn't become another burdensome legacy system in the near future.

There are five main ways that these compromises manifest in change projects:

1. The change project is viewed as an "IT thing"
2. A digital skills gap
3. Budgetary constraints
4. Time pressure
5. Trusted partners not acting in the long-term interests of the business

1. The change project is viewed as an “IT thing”

Over the last few decades, Thoughtworks has partnered with a number of leading organizations to help them drive digital change initiatives. The first major challenge we’ve encountered in many of them is that enterprise leadership treats these change programs as an “IT/technology thing”.

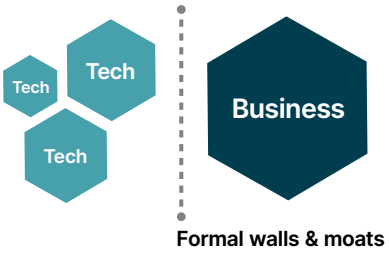

Figure 1. Supporting role: Tech creates digital assets

One of the biggest drivers for this perception is that IT is still viewed as a support function by top management. They’re seen as the department that enables the rest of the business to deliver ambitious technology-driven changes and is responsible for their planning and delivery, and kept siloed away from the rest of the business as shown in the diagram above.

For change projects to be successful, cross-functional collaboration is essential. IT and line of business teams must work together to devise and implement practical, usable solutions for customers. Without that collaboration, the project outcome will almost certainly be riddled with accidental complexity and will likely become a contributor to the legacy estate.

Changing entrenched attitudes and culture isn’t simple. If you recognize similar challenges in your organization, possible avenues to explore are in executive coaching, increasing transparency and demonstrating value delivery

to the business as early and frequently as possible to show the intrinsic link between IT and business success.

2. A digital skills gap

Running a successful change program requires experience, dedication and specialist digital skills. Unfortunately, building a team of people with all of those qualities — and a genuine desire to drive your planned change — isn't always easy.

The easiest way to fill digital skills gaps — especially for skills that are only needed short-term — is to simply bring new people in to help manage and execute your planned project. However, this brings with it a whole host of new challenges that can lead to accidental complexity.

When you've got new people coming from different company cultures and backgrounds, they face a core DNA mismatch with the organization they're changing. When things need realigning, that turns into an 'us versus them' issue, ultimately meaning that key issues go unresolved, and systems launch without essential capabilities.

In many cases, inexperienced external leaders adopt a "power through" attitude, where their focus becomes delivering the project to the original scope, rather than course-correcting and making difficult decisions to deliver the best end result.

Tackling this challenge takes time. But, by ensuring you work with a team that has the right digital skills and capabilities — and more importantly, the right attitude towards change — you can make it happen.

Choose people that have the skills you need to bring digital change to life, but also ensure that they are truly invested in the long-term organizational impact of that project. Ensure that any specialists drafted in from outside the organization are experienced and well-reputed in driving such change programs.

3. Budgetary constraints

It doesn't take long for most change teams to realize that initial budgetary estimates don't align with what's actually required to deliver the project. In the end, initial estimates are exactly that — best guesses made on past experience and rough predictions.

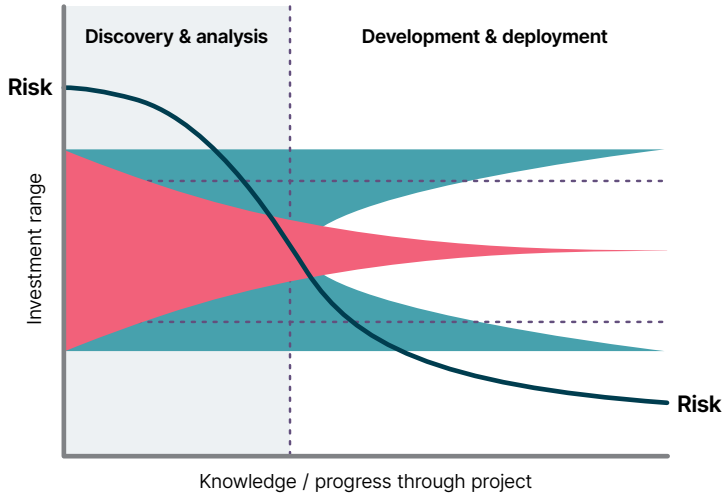

Figure 2. Cone of uncertainty

A simple cone of uncertainty exercise (shown above) can reveal this fact. However, many teams are ironically asked to commit to the deadlines and the budget based on these estimates.

Teams are expected to work on the basis of these initial estimates and operate within a boundary based purely on speculative outlines. Extending budgets is viewed as project failure, when more often than not, it's requested because the team has recognized a need for essential additional work — things that nobody could have forecasted before the project began.

Consider this slightly reductive analogy. An archaeological team has found the first signs of a particularly interesting dinosaur skeleton buried at a new dig site. Using the information and survey equipment available to them, they estimate that they'll need $50,000 to complete the excavation project. During the excavation however, it becomes clear that the mass of bone identified is actually two skeletons — one slightly below the other. The scope of the project changes immediately and additional budget is required to recover the second skeleton successfully.

It seems only logical that the people funding that project would be willing to extend the budget and accommodate this new scope. After all, recovering dinosaur bones is their core mission. However, in IT projects, this often isn't how that discussion goes.

Despite budget changes being requested for good reason — reasons that will help ensure the project delivers its intended outcomes and business value — teams are expected to simply adapt and operate within the originally scoped outline. Soon, the team finds itself developing loose or 'duct tape' solutions to the real engineering challenges at hand, simply doing their best with what they're given. This is the beginning of a very large accumulation of technical debt.

Tackling this issue is quite difficult, because it requires the wider organization to change in many ways. However, a courageous leader could look at options like Beyond Budgeting, Bossa Nova, Virtual Invoices, VC Funding Models, and Cost of Delay exercises.

One crucial point here is the importance of detailed inception and discovery exercises. They cost very little compared to the change programs themselves and they can significantly improve estimation accuracy.

4. Time pressure

Everything that we've just said about budgetary constraints can also be applied to project timelines. They're estimated in the same speculative way and any valuable extensions to the original project scope will always invalidate them.

We cannot emphasise enough that in almost all major IT change projects I've witnessed, these deadlines aren't based on any real justifiable business logic. They're chosen arbitrarily, based on rough project-scoping exercises and quickly become rigid, inflexible targets that the team is often forced to stick to for no real reason other than pleasing some senior stakeholders.

Teams working under this constraint often run into various collaboration and dependency challenges. The decisions made under such pressures are mostly loose and the results are almost never satisfactory. It also fuels a culture of blame when things start going wrong. Together, those factors make this a huge contributor to complexity and the growth of the legacy estate.

Just like stability concerns, most deadlines are nothing but an illusion. A smart leader will look deeper to understand exactly when each individual aspect of a change project is needed by the business. They don't try to deliver the whole, they deliver what's actually needed. They'll prioritize the small deliverables with well-estimated time limits, and keep the rest of the project flexible to ensure the right outcomes can be delivered.

5. Trusted partners not acting in the long-term interests of the business

Many enterprises that initiate big change programs tend to bring in partners to help them on their journey. While this is a very valid, and often necessary choice, it can cause organizational inertia and loss of control within the program if it's not executed carefully.

Without full accountability, the solutions that partners create can contribute to complexity and the legacy estate by:

- Being fundamentally disconnected from the culture of the organization, so users never fully embrace them
- Solving an immediate problem but leading to integration and change barriers in the future
- Forcing projects through against tight timelines, ignoring changes and revisions that should be made to improve outcomes

It's vital to recognize the strength of the link between cultural and technological change. Solutions and change projects cannot simply solve a current business challenge on paper; they must do it in a way that works for the end users, for whom it will ultimately become a key part of their working life.

When a partner is brought in to help execute a change project, they need to deliver not only what the business wants from a solution now, but what its users and IT team will be able to effectively make use of, long into the future. That means focusing clearly on customer and developer experience.

The DNA of almost all change programs in technology heavily relies on capability development and the build-up of organizational muscle. Having the right organisational mechanics, and the right partner with you on this journey — one that's been there multiple times before — will give you stronger protection against accidental complexities that create a future legacy estate.

The impact of insufficient care and maintenance

The impact of insufficient care and maintenance

Care and maintenance keep systems healthy and operational during their lifetime. But, as soon as they stop, systems can quickly become a tangled mess of complexity that's hard to manage and upgrade — condemning even the best systems to become part of the legacy estate.

Across most enterprises, there are five main reasons why system care and maintenance break down:

1. A lack of visibility
2. A lack of timely investment
3. A lack of percieved business value
4. Loss of essential skills and knowhow
5. The gradual accumulation of technical debt

1. A lack of visibility

Systems will change rapidly over time and one of the most common reasons they're not properly maintained is that people simply can't see how their maintenance needs have changed.

With systems constantly evolving alongside numerous business, environmental and technological parameters, planning for future maintenance needs is extremely challenging and cannot be done beyond a visible timeframe. You can't expect teams to plan out future care and maintenance months and years in advance. But, what you can do is ensure that the people responsible for maintaining systems stay aligned with the overall goals and vision of the business. If systems are continuously being

cared for and maintained with the current goals and vision for the business in mind, accidental complexity can generally be avoided.

One tool that can help you here is the Lean Value Tree (LVT). Among its many benefits, it can help you build a common understanding of the business' goals, strategy and vision by simply visualizing them in the simplest way for your entire staff. Then, everyone is empowered to act in a way that best helps the business achieve those long-term goals.

The LVT is an effective visualization tool that helps an organization see where they are going, what decisions have been made and how fast they are progressing. To enable teams to make good choices and measure their performance, it's important to know what measurements demonstrate success.

Each box in the LVT is a call to action. As a tree-shaped artifact, everything stems from a clear vision of the future. Each aspect of the tree contributes to the accomplishment of the leading vision through experiments. Multiple goals, bets and initiatives create a strategy based on testing, where failure can happen and drive new attempts.

Measures of success cascade upwards and are aligned to support the goal. Bet and initiative owners should evaluate goal success measures and decide if there's a more specific measure related to the specific bet or initiative. Measures of success should beoutcomes focused as shown in the example below.

Strategic alignment around clear measures of value leads to greater responsiveness and customer-centricity. It produces a more effective organization. As a beneficial side-effect, it also

Figure 3. Cascading layers of success in a Lean Value Tree

produces a more efficient one by questioning established pots of budget that aren't directly related to the customer-driven purpose.

2. A lack of timely investment

It's very rare that opportunities to invest in the improvement or transformation of an existing system occur organically. Investment budgets are typically reserved for exciting new projects and the creation of entirely new systems. And if an

existing system isn't seen as supporting the current vision of the business, it has practically no chance of receiving valuable maintenance work.

However, despite that, enterprises all around the world seem content to spend millions just to keep these systems running. They want to "keep the lights on", and they're spending a fortune to do so but they're completely against investing a fraction of that money to swap to more energy-efficient light bulbs.

Over time, the list of small but essential updates and maintenance tasks that could be implemented for existing systems grows, until the pile becomes so huge and carries such a high cost there's no chance of the organization investing in it further. The result? A system that's too critical to lose, too big to move and too encumbered by legacy demands to modernize. That's a monolith.

The only real way to prevent this is to change your attitude towards investment in existing systems. Smaller update and maintenance tasks should never be avoided — to prevent them from piling up — and should just be considered a cost of keeping the system operational and efficient.

However, it's also important to think critically about which upgrades or new extensions to existing systems are truly needed. If an add-on, plugin installation or a proposed maintenance task can't clearly be linked to business value, it can likely be avoided. Applying this critical judgement to new tasks can also help keep your backlog clear and prevent the build up of maintenance debt.

3. A lack of perceived business value

It is almost paradoxical that nobody wants an inefficient system and yet when it comes to putting an investment forward to make the system efficient and optimal again, it becomes difficult to justify the ROI of the change required.

Many senior leaders struggle to come up with a business case that justifies the change. The change requests are often viewed as an IT expense and quickly disregarded because they simply don't generate the same level of business interest and perceived benefit as net new investments and projects.

The result is that the IT team ends up burdened by legacy technology that it simply isn't given the resources to improve. It's a problem experienced by almost every major enterprise — and the only way to change it is by changing the way investments in existing systems are framed.

IT teams, and the leaders that oversee them, need to start building stronger business cases for maintenance investments — framed around immediate business needs such as regulatory change or shifts in tax or risk demands. To generate relevant interest and display appropriately appealing ROI, those cases should make it clear how a proposed maintenance project will:

- Help the existing system better meet the current demands and direction of the business — at a lower cost than creating a new system
- Help prevent the kinds of high-profile disasters caused by legacy IT that we've seen dominate headlines in recent years — particularly those relating to data loss or service outages

- Reduce the overall management burden associated with the existing system, and enable the system to recoup the investment itself through lower upkeep costs

This is where you could really use Business Capability Mapping, Cost of Delay, and Lean Value Tree exercises to come up with the right business value definition for the desired change.

4. Loss of essential skills and knowhow

Perhaps the most widely acknowledged reason that existing systems stop being maintained and become legacy is because the people that understand how they work have left the organization. In IT, times change — approaches, hardware, processes and even programming languages fall out of favor, and pretty soon it becomes hard to find the right people with the skills to maintain systems built on them.

The core problem here is that most organizations aren't very good at managing and maintaining knowledge — so they can't take the right actions to ensure essential capabilities remain in the business.

Right now, the problem can be broken down into three aspects:

a. **Visibility:** In many organizations, there's no enterprise knowledge map. In the absence of this map, it's quite difficult

to answer who needs a shadow, what kind of training is needed, or what the right staffing and hiring plans are to ensure that impending gaps are filled proactively.

b. **Process:** Most enterprises don't have any formalized knowledge transfer processes — let alone ones that can be adapted at speed. A good litmus test for how strong yours are is when you hire a new software developer, how fast can they actually make a change in the software and check that into the company SCM? Is it in hours, days, or weeks?

c. **Governance:** A lack of governance around information capture and transfers often leads to wrong knowledge acquisition, invalid or out-of-date information being passed on, and more operational mistakes.

Ultimately, any solution devised to solve this issue will need to be tailored to the culture of your organization. It's all about finding what works best for your teams, then putting the right governance in place to make sure that timely knowledge transfer is actually happening — and that it's happening well.

5. The gradual accumulation of technical debt

This is closely linked to all the causes of insufficient maintenance we've just explored. Over time, small instances of these causes slowly accumulate and combine to turn current systems into legacy infrastructure.

The important point here is that this is the underlying cause behind almost all legacy systems in operation at the moment. There is no single event that makes them become legacy

systems — it's a slow, creeping decay that happens as the multiple causes pile up over time.

Small issues and maintenance requests go unanswered, while new integrations and workflows continue to pour into the system. Pretty soon, you've got a mess on your hands that can't be untangled — the kind of legacy system that I'm sure you can relate to.

The only practical solution is to stay on top of all the small issues, before they get a chance to pile up. That's the only time you can effectively combat them — when they're current, small and limited in number. To do that, once again you need to change your culture and empower teams to tackle each of these issues head-on.

A gradual piling up of issues wouldn't be tolerated in any other function of a modern business — so why do we still tolerate it in IT?

Summary

Summary

Nobody intentionally creates legacy systems. Every one of them was implemented for a reason, and every decision made regarding their upkeep and operation has been made for a reason too.

In some cases, the build-up of legacy complexity can be entirely unavoidable, giving rise to a monolith that doesn't actually need to be broken. However, this is rare. In most cases, legacy complexity happens by accident, slowly creeping into efficient systems and turning them into lumbering hindrances to the organization and the IT team.

To truly break the monolith, business and IT leaders must refocus on what 'the monolith' really is. It isn't an individual inflexible system that's blocking transformation. It's the processes, culture, patterns of behavior and attitudes to change projects and maintenance that surround those systems.

Some of those patterns and cultures are very visible. Others are harder to surface. But, to effectively improve an organization's agility and its ability to change in the long term, they must all be carefully examined and recalibrated to better support the business.

It's a daunting task, but it's far from impossible. It demands strong leadership — in both the business and the IT function — and requires true passion and commitment to positive change. But, those that can challenge entrenched ways of thinking and root out the underlying causes of accidental complexity stand to gain almost immeasurable long-term returns.

With as much as 80% of an IT team's time spent maintaining legacy monoliths, breaking them down will enable you to build something far greater in their place — and completely transform the way you approach and manage IT for decades to come.

About the author

Isa Goksu

Director of Technology, Financial Services, Thoughtworks UK

Isa is the head of technology for Financial Services practice at Thoughtworks UK. He is a banking and finance domain expert, and a passionate technologist with over 20 years of consulting experience. He has spearheaded several challenging digital transformation initiatives while managing globally distributed delivery teams. He is an expert at agile engineering practices and has a technology-agnostic approach to solving enterprise-wide change problems. He has developed AI-based market-making algorithms for trading and is passionate about machine learning and distributed systems.

He has led major technology transformations while working alongside CIOs at many large UK, European and North American Financial Services clients. He was instrumental in building the core trading engine at one of the world's largest stock exchange.

You can reach him on **isa@thoughtworks.com** or connect with him on **LinkedIn**.

Get in touch with us at

contact-uk@thoughtworks.com

Thoughtworks Ltd.
First Floor, 76-78 Wardour Street
London, W1F 0UR, UK
+44 (0)20 3437 0990
thoughtworks.com

/thoughtworks