

THE DEFINITIVE TECHNICAL DEBT SELF-ASSESSMENT

*Determine The Depth Of Your Technical Debt
So You Can Build A Plan To Get Out Of It*

Contents

3	About this Assessment
4	How to Fill Out the Assessment
5	Your Goals
6	How is Technical Debt Sucking the Soul Out of Your Teams?
7	What is Technical Debt Costing Your Organization and Business?
8	Your Analysis of the Cost of Technical Debt to Your Organization
9	One More Thing: What Happens If You Do Not Fix Your Technical Debt?
10	Which of The 7 Types of Technical Debt Do You Have?
18	Other Types of Technical Debt?
19	The Costs of Tech Debt Revisited
20	The Five Keys You Need to Have a Lasting Victory over Technical Debt
26	There is a Way Out!
27	No Assessment is Complete Without Taking Action
28	About Exceptional Difference

About this Assessment

This assessment guide will do the following key things for you.

- 1 You will be able to make a quick but thorough assessment of how deep you and your teams are mired in technical debt
- 2 You will be shown the five keys that teams have used to thoroughly defeat the technical debt holes they found themselves in
- 3 You will finish the assessment with a number of ideas of what you can do to address your technical debt issues
- 4 You will have your dream restored of achieving the promise of amazing software engineering

First, a very brief history of the word “technical debt.” The term originated in 1992, when Ward Cunningham uses a metaphor to explain (to non-technical stakeholders) why resources needed to be budgeted for refactoring.. “With borrowed money, you can do something sooner than you might otherwise, but then until you pay back that money you’ll be paying interest. Rushing a product out the door to get some experience is a good idea, but you eventually repay that loan by refactoring to reflect your experience.” Over the next three decades, the concept of technical debt expanded to mean “any of the things in the development life cycle that slow you down.” (We told you it is a very brief history.)

This assessment was built based on our work with thousands of engineers worldwide. It is organized into a few major sections based on the patterns we have seen where the best teams have thoroughly vanquished their technical debt issues.

GOALS	COSTS	TYPES OF DEBT	THE WAY OUT OF DEBT
What are your goals regarding technical debt? Yes, your goal is not to have any. However, the best teams have clear goals about why that is important to them. Take a few minutes to think about why it is important to you.	You will assess the costs of technical debt to your team’s spirit. (Yes, we said spirit. That is just as important as economics!) You will also make an estimate of the tangible costs of technical debt to the business.	Did you know there are 7 types of technical debt? You will be provided a description of each of the 7 types and guided to assess their impact on you and your organization. You may be surprised by some debt items you have just taken for “business as usual.”	You will be guided through <i>The Five Keys You Need to Have a Lasting Victory over Technical Debt</i> . You will assess yourself and your teams against how well you are doing with each of these keys. That will lead you into the final section, where you begin to outline a plan to overcome debt for you and your organization.

How to Fill Out the Assessment

You can type your answers in the open spaces directly in the PDF. Some people prefer to print it out and write in the answers. Some people find it useful to have another document open to record more in-depth ideas. Some people use a notebook to write in many ideas and keep adding more ideas as they go about their day. The choice is yours.

Our main guidance is to consider doing a two-pass process. Go through it once and do quick off the top of your head answers. Go through a second time and you will likely add more depth to your answers.

Enjoy the journey!

If You Need Help, We Are Here

During the course of filling out this guide you may find yourself in need of a little help. Don't hesitate to reach out to our team for assistance in this endeavor. For decades we have helped thousands of engineers and team leaders think through the challenges associated with technical debt and other engineering problems. Our mission here at Exceptional Difference is to change the world of engineering for the better so you can do the work you love.

Just email info@exceptionaldifference.com or call this number (607) 269-8984 to connect.



Julia Mullaney, CEO



Allan Willett, Chief Engineer

Your Goals



No one likes being slowed down by technical debt.

In this section, consider your goals. If you could wave a magic wand and all the technical debt could disappear, what would that enable you to do?

Define your goals as specific outcomes. Consider this generic goal as an example: ***we would be able to build a new product line to satisfy a new market segment.***

Go ahead, write down your big wishes. Articulate them as specific and tangible goals.

GOAL #1

GOAL #2

GOAL #3

How is Technical Debt Sucking the Soul Out of Your Teams?

Do You Spend Your Time in “Create” Or “Waste”?

The technical debt problem is vividly represented in this picture. The green in the picture is the creative side of the work that engineers love to do. It is figuring out the real user needs through user stories. It is building designs to solve complex challenges. It is writing the code that will implement solutions.

The red in this picture is waste. The effort does not add to the value side of the equation. This is time lost to doing rework.

At Exceptional Difference, we have worked with 1000s of people working in various domains, including financial services, space projects, defense projects, information technology companies, consumer sales systems, navigation, and more.

We studied where engineers spent their time in these companies.

And although each team is unique in their work and how they do it, we found that all the teams share these technical debt problems.



On average, these companies lost 65% of their development time. In other words, their development teams lost all that time to rework and technical debt issues.

Before we look at the economic cost, consider the cost of your team’s energy. Which of these is true for you and your team? Check all those that apply:

Feeling Frustrated

Working late to overcome what should have been much easier

Feeling plagued by tons of little (and big problems)

Having to deal too often with “spooky motion at a distance,” where you make a change in one part and that mysteriously breaks something in another part of the system

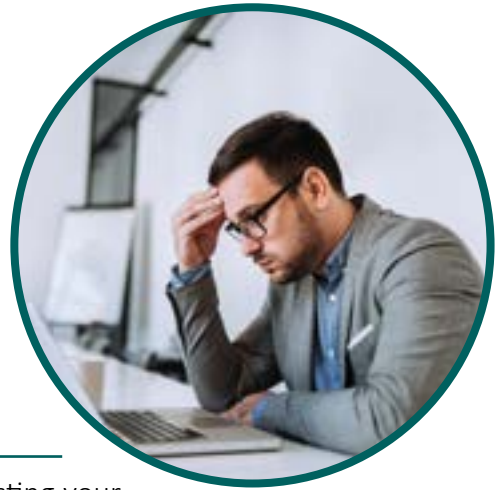
Making it work because there was no time for a re-design

The business side doesn’t want us wasting time fixing things that already “work”

The business side keeps asking for more complex features

Feeling like most of the time, we are trying to crawl up a muddy hill while people push boulders from the top at us

What is Technical Debt Costing Your Organization and Business?



The key challenge of this page is to consider what is technical debt costing your organization. What is technical costing the business?

When you take on financial debt, you pay back that debt with interest. That interest is paid in money.

When you have technical debt, it costs you, your team, and your organization something as well. That something is lost time.

To get started with a plan for addressing technical debt, it is important to first consider how much time the technical debt is costing you and what it will take to pay it off now so you can stop paying with lost time week after week, month after month, and year after year.

Consider this example from one of the teams we worked with:

Pat's team analyzed an item to determine that it would take only three days to refactor that item completely. That sounds small, but it was a piece of *debt that all developers had complained about over the last three years*. They determined that every time they touched this item, it cost them an additional five days because of that tech debt. Five days is no big deal. Why bother?

But, the team typically had to change this item 30 times a year.

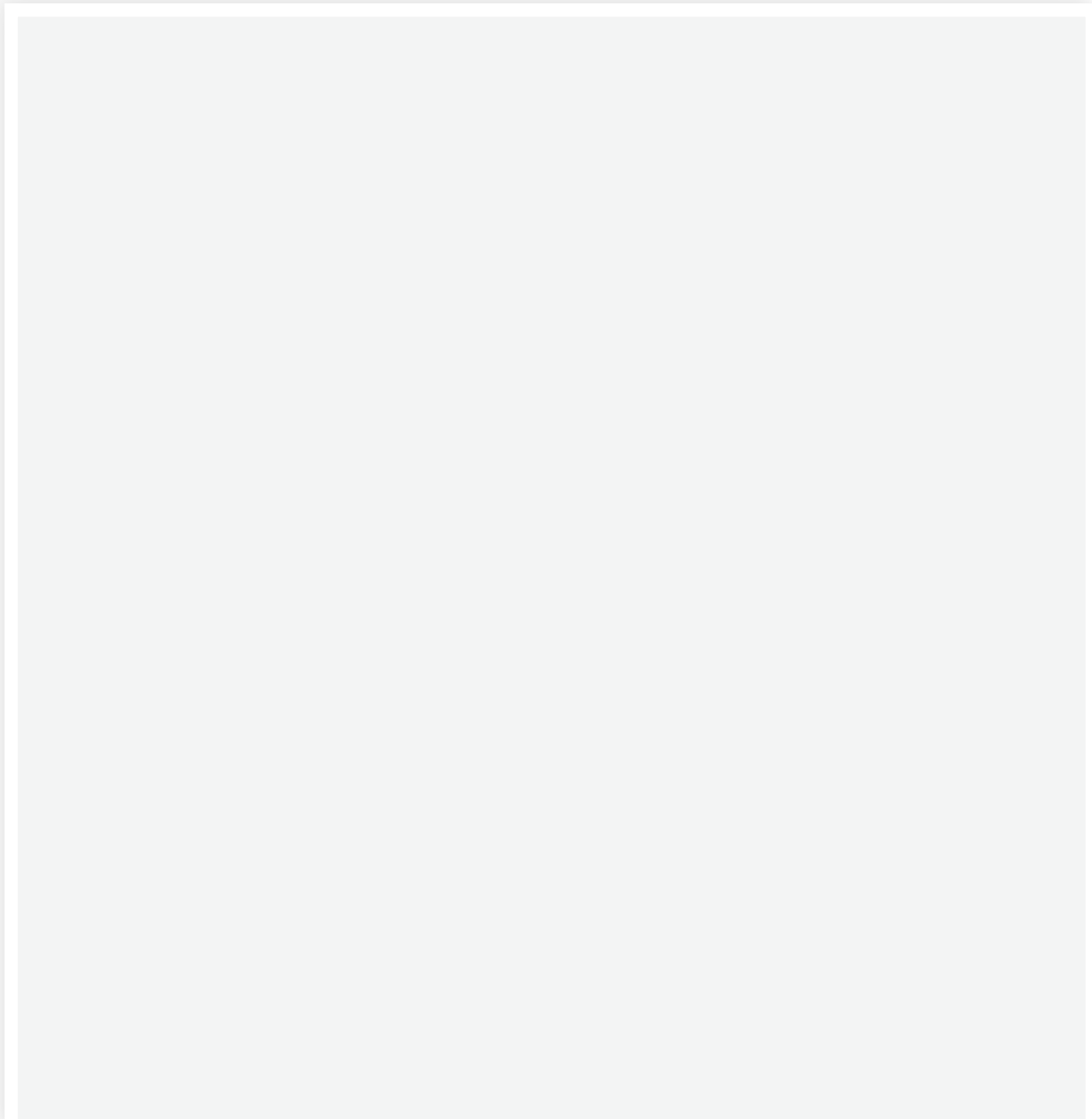
In three days, they redesigned the item to removed that debt and discovered hey were wrong. It saved them seven days of effort every time they had to work in that item. ***With three days of effort, they found they had saved 210 days for the following year.*** They were excited because they knew they could do amazing things with that time saved.

The ROI on this effort was incredible. And there are many other examples. Some even more dramatic in the amount of engineering time saved. One team invested six people months in fixing a Gordian Knot (explained later in this guide), ***saving two years of time every year thereafter.***

On the next page, left blank for you, take some time to do your own back-of-the-envelope estimates. This can be challenging without experience. **Remember, if you need help, just schedule time with Exceptional Difference for a free strategic assessment session: info@exceptionaldifference.com**

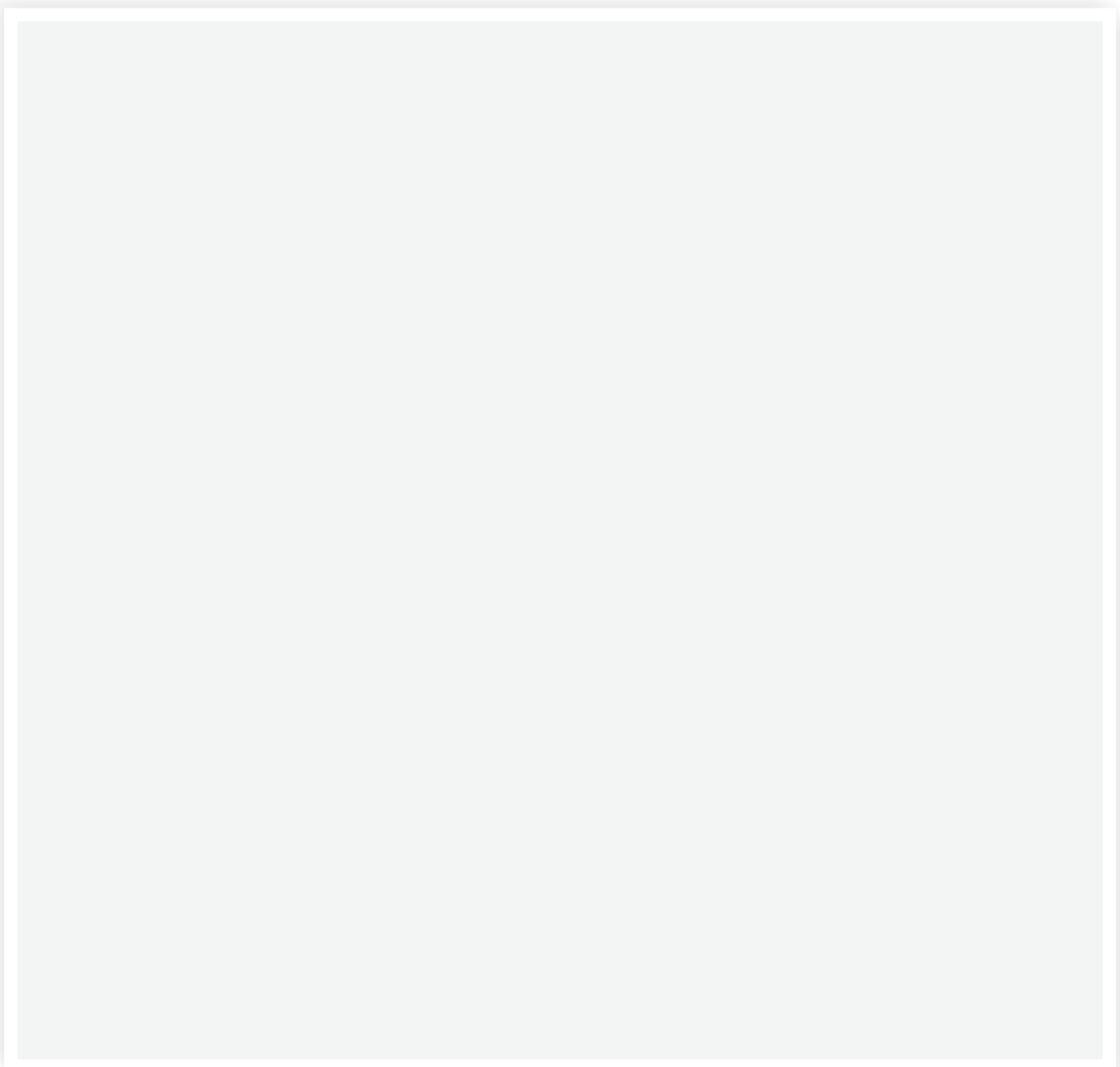
Your Analysis of the Cost of Technical Debt to Your Organization

This page is for you to make notes about the cost of technical debt to you and your organization.



One More Thing: What Happens if You Do Not Fix Your Technical Debt?

Bad things happen when organizations keep incurring more technical debt without fixing it. They don't just get slower. Their products start losing value in the eyes of their customers. And sometimes, they fall off the technical debt cliff. What will happen to your organization and your products?



Which of the 7 Types of Technical Debt Do You Have?

The following is a quick overview of the major types of things that slow down many organizations. Read through this overview, and we will walk you through some questions about each type. Doing this will help you better understand the technical debt you and your team are dealing with.



1. Gordian Knots

Implementations have highly convoluted and complex structures, like balls of spaghetti. Developers fear making changes because it is hard to make changes and easy to break things.

2. Barnacle Ware

Implementations start with core features (the boat) that work well. Various customers each ask for a special feature (a barnacle) just for them. Soon there are more barnacles than boat.



3. Glut

When the system is choked with too many low-value items. This glut slows development and system performance. Examples include feature glut, application glut, and data glut.

4. Environmental Drag

Work feels like a difficult slog through thick mud. Cumbersome tools, low-value process steps, bad meetings, and other numerous small annoyances accumulate and create drags on speed.



5. The World Moved On...

Developers feel trapped in the past even as they see “better-faster” all around them. Examples include systems that rely on out-of-date technology, or underlying designs that do not match current approaches.

6. Trap Doors

Work that was moving forward well gets derailed by a trap door opening. For example, the foundation on which the system depends has incomplete, incorrect, or even missing parts. Changes that should have been quick end up taking a long time.



7. Skill Debt

The lack of experience in the critical areas that slows everything down. New customer or technology domains can cause experienced people to be rookies. More people could help, but mainly an expertise infusion is needed.



1. Gordian Knots

We call implementations with highly convoluted and complex structures, like spaghetti balls, Gordian knots. These are intractable problems that can only be solved by cutting the knot, just as Alexander the Great did to the original Gordian knot. Developers fear these parts of the system because making changes is hard and changes often break other parts of the system. This often happens to a core piece of the system. It worked well when it was initially designed, but over time new needs arose and many changes were made. Developers cringe when they have to make more changes inside the mess.

On this page, create a list of the gordian knots that exist in your system. Consider how big they are. Consider what types of pain they cause. Contemplate how much pain each one of them is causing you and your teams.



2. Barnacle Ware

Implementations start with core features (the boat) that work well. Various customers each ask for a special feature (a barnacle) just for them. Soon there are more barnacles than boat. When the barnacles start costing you speed, you have a problem.

In some companies, we have seen the customer *specials* (the barnacles) become larger than the core system itself. Furthering the pain, it often turns out that those barnacles eat up engineers' time, but only bring in a small percentage of the revenue.

On this page, create a list of the barnacles that exist in your system. Consider how big they are. Consider what types of pain they cause. Contemplate how much pain each one of them is causing you and your teams.



3. Glut

Glut is when the system is choked with too many low-value items. This glut slows development and system performance. Here are three examples of Glut.

- **Feature Glut**

- Let's first consider the user's viewpoint. Have you ever used an application that was fun and easy to use? And then, over time, more and more features were added. Did you notice the first ones you liked, but soon the application seemed crowded and harder to use? Has your team created that for some of your users?
- Now consider it from the developer's viewpoint. Are you maintaining several features that very few customers ever use? Does the maintenance of those features slow you down? Are you maintaining any pieces of the system that are never used by anyone at all?

- **Application Glut.** Many IT teams find they are maintaining *hundreds* of apps. Many are redundant or overlap. Some only have a handful of users. Is this slowing your team down?

- **Data Glut.** We have seen customer databases with millions of entries where 50% of the entries were from companies that no longer exist. This slowed down every interaction with the database. This led to erroneous analysis. Do you have any examples of data glut?

On this page, list types of glut that exist in your system and the impacts these items have on your team.



4. Environmental Drag

Work often feels like a difficult slog through thick mud. The environment slows team down due to things like cumbersome tools, low-value process steps, and low-value meetings. People become so used to the environmental mud they don't even consider that many of these items can be addressed. These small annoyances accumulate and create significant drags on speed.

What environmental drags on speed does your organization experience?

A large, empty rectangular box with a light gray background, intended for users to write their responses to the question above.



5. The World Moves On

Developers feel trapped in the past even as they see “better-faster” all around them.

One example is systems that rely on out-of-date technology. More than one organization we worked with was relying on compilers where the company that created the compiler was out of business. Another example is where the original system designs do not match current approaches the developers are using. The conflict between the approaches causes a significant drag on speed and hinders quality development.

Many times these companies are at risk of going over the technical debt cliff.

On this page, list types of “the world moves on” debt that exist in your system and the impacts these items have on your team.

A large, empty rectangular box with a light gray background, intended for listing types of "the world moves on" debt and their impacts.



6. Trap Doors

Work that was moving forward often gets derailed by a trap door opening. For example, the system's foundation has incomplete, incorrect, or even missing parts. Changes that should have been quick end up taking a long time.

On this page list types of "trap door" debt that exist in your system and the impacts these items have on your team.

A large, empty rectangular box with a light gray background, intended for users to list types of "trap door" debt and their impacts. The box is framed by a thin white border and has a subtle drop shadow.



7. Skill Debt

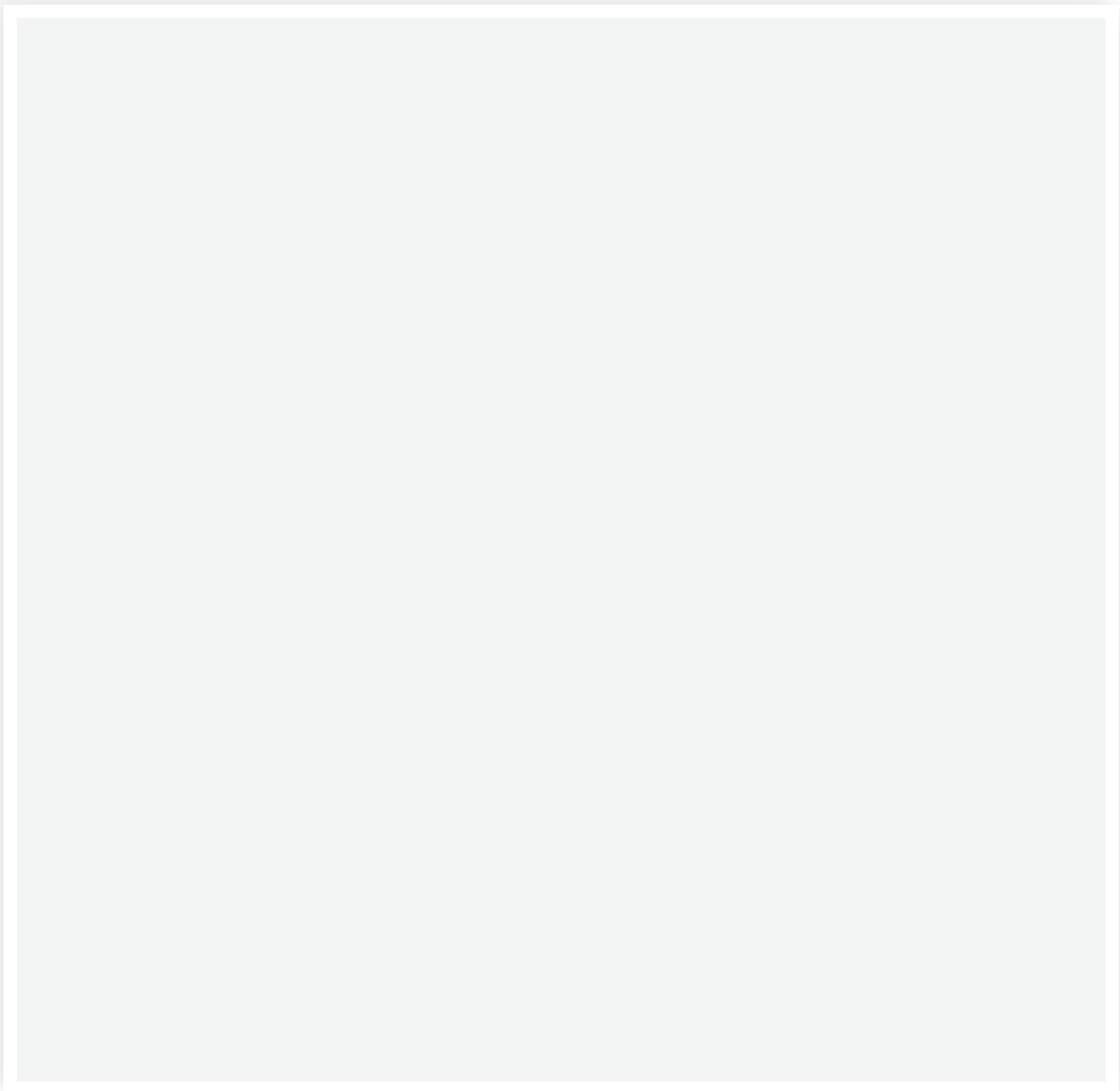
The lack of experience in critical areas slows everything down. It helps if your people are extremely talented. But, a new customer domain or technology domain can cause experienced people to be rookies. Ironically, sometimes when organizations fix other types of debt (such as *the world moved on* with new technology, or environmental drag with new tools), they suddenly find themselves with skill debt. An expertise infusion is needed to accelerate the learning and thus the speed to developing customer value.

On this page, list any skill debt area that exists in your system and the impacts these items have on your team.

A large, empty rectangular box with a light gray background, intended for users to list skill debt areas and their impacts. The box is framed by a white border and is positioned below the introductory text.

Other Types of Technical Debt?

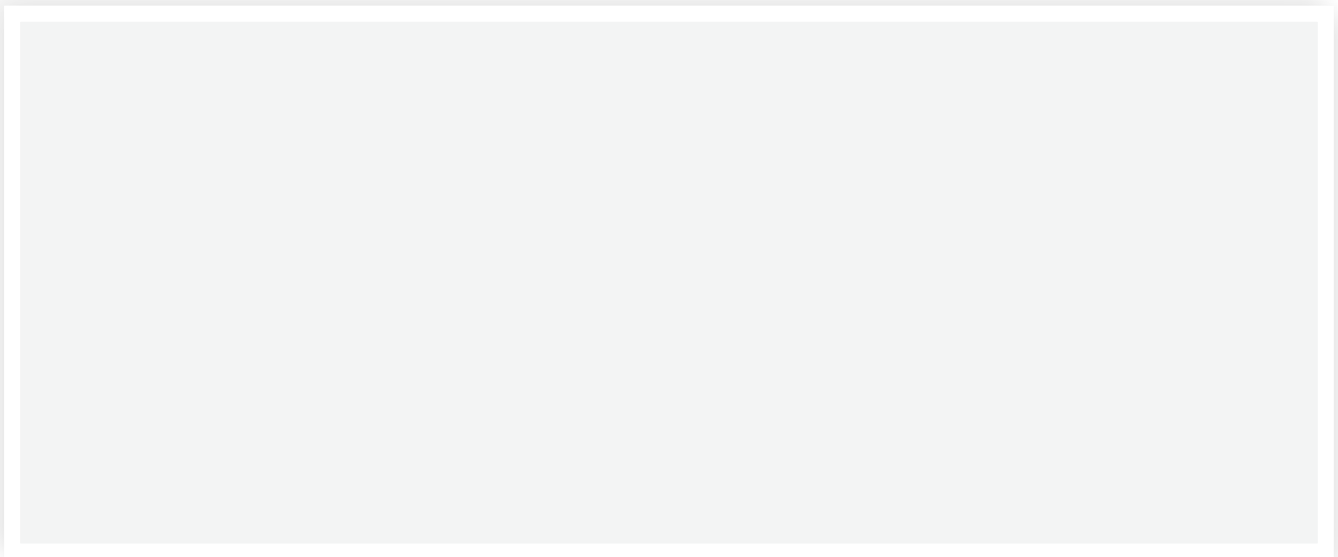
When we talk about the 7 types of technical debt, people often mention other types they have experienced. What special types of drags on speed does your organization experience?



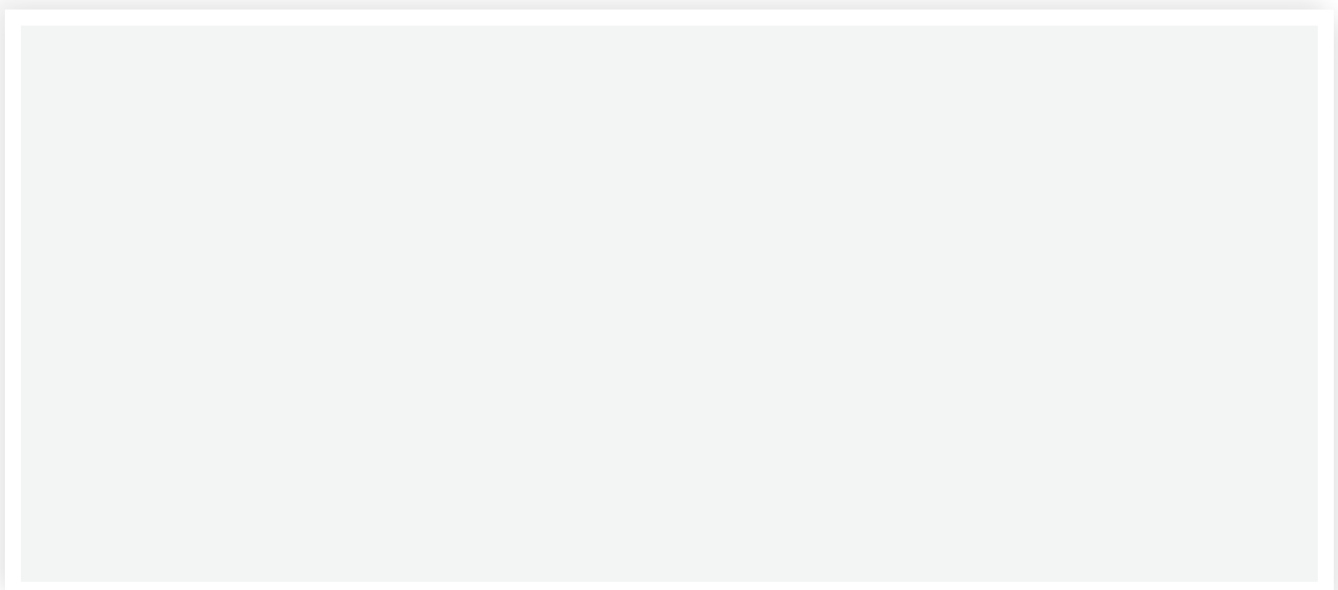
The Costs of Tech Debt Revisited

In our experience, when people finish doing the self-assessment of the 7 types of technical debt, they find the problem is a bit worse than they expected.

Revisit your first answer about the tech debt cost to your organization. How does your original response change from the prior “soul-crushing” section? Update and add notes here.



How does your answer change, considering all the costs associated with all types of technical debt? Go back to that page, add updates, or add some notes here.



The Five Keys to Having Lasting Victory Over Technical Debt

From working with thousands of teams around the world, we have found that successful teams have the following five keys to success. With these keys, teams can achieve amazing, jaw-dropping improvements.

Key 1: Tech Debt Maps

Without a map, you don't know where you are.

A technical debt map illustrates the debt in the entire system in an easy-to-visualize format. The areas of technical debt are highlighted with a color-coded scheme that clearly shows the hot spots. Beneath the map is a sophisticated system based on objective and subjective measures that determine the colors on the map.

The maps enable engineers to focus on the most important areas to work on next. The maps enable managers and product owners to see progress over time. Most of all, it provides a context for the whole organization to have constructive conversations about technical debt.

- What kind of technical debt maps does your organization have?
- How would you rate the effectiveness of your organizational technical debt maps?
 - None exist
 - We created one once but haven't looked again
 - Oh, they exist, but they are more busy work than useful and used
 - We have them, but they need work to truly make an impact
 - Our maps are works of beauty. It is a sea of green (where green is good), and we keep it constantly updated
- Note any actions you should take to make improvements in this area.

Key 2: Tech Debt ROI Calculator

How do you know what tech debt items are worth fixing? How do you know the order in which they should be addressed?

The best teams have a quick yet thorough way to determine the return on investment (ROI) for fixing technical debt. They can ascertain and quantify the costs being incurred by carrying the debt. They can accurately estimate the effort required to remove the debt.

These teams can augment their technical debt maps with ROI data which enables the organization to agree on a investment plan.

■ How would you rate the effectiveness of your Technical Debt ROI Calculator?

None exist

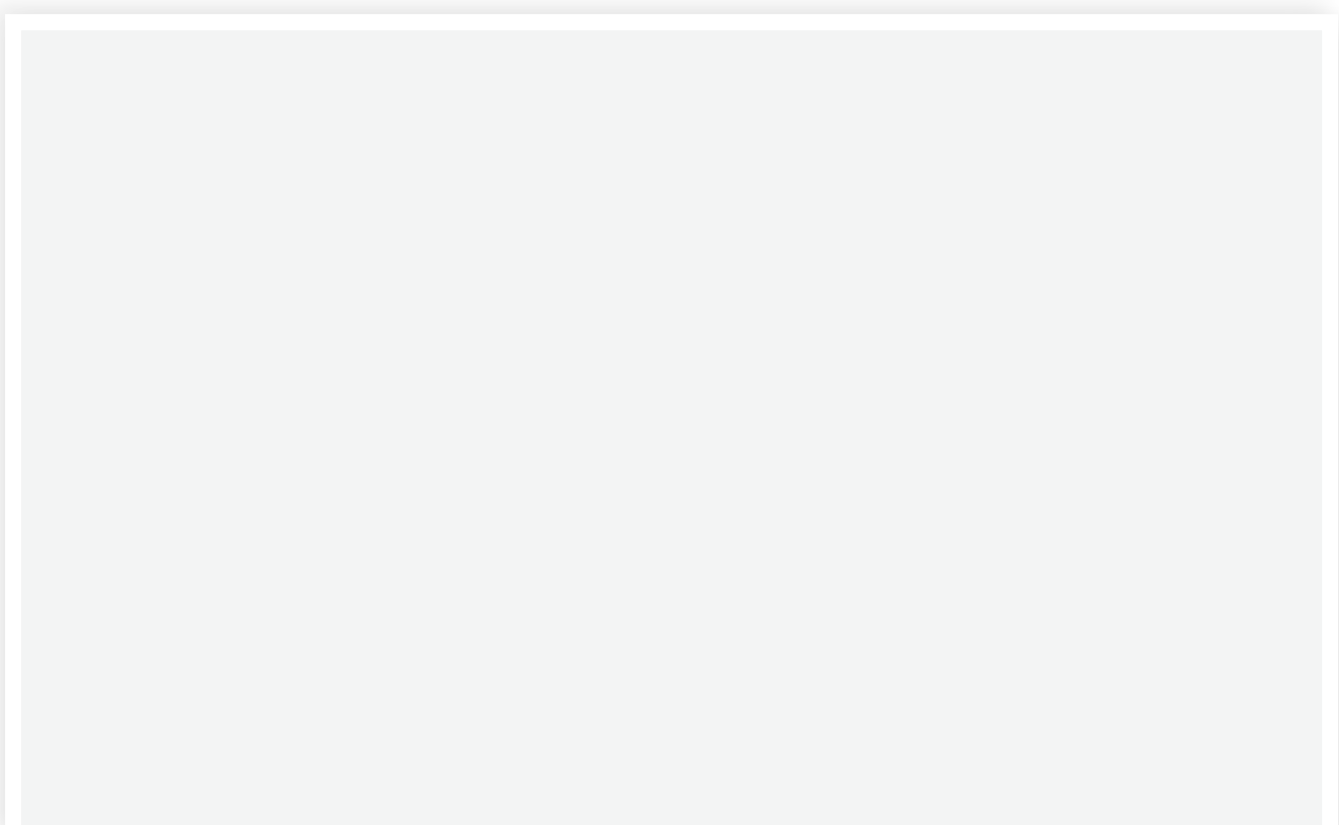
We wish we had that, but that is more work: “we don’t have the time”

We tried that once but realized, “we don’t need any stinking data”

We have them, but they need work to be used and useful

Yes, we have that, and it works wonderfully

■ Note any actions you should take to make improvements in this area.



Key 3: Early Warning Systems

The cars we drive have excellent early warning systems. The fuel meter tells us how much gas is in the car. A light comes on and a chime sounds when the car hits low fuel. Lights and sounds come on for various other reasons to keep your car in maximum good health.

Organizations that manage technical debt effectively have early warning systems in place. These systems employ objective, customized measures as part of their toolsets. For example, the best organizations we have worked with don't allow a cyclomatic complexity above 10 without a signed (electronically, of course) waiver.

The best systems make the engineers aware of the cost of technical debt. Additionally, these organization use subjective warning systems as well. For example, any team member can raise a technical debt yellow (or even red) flag when they see something going in the wrong direction. This could be any one of the types of technical debt, from a software module deviating from the design pattern to a process step becoming valueless.

- How would you rate the effectiveness of your Early Warning System?

None exist

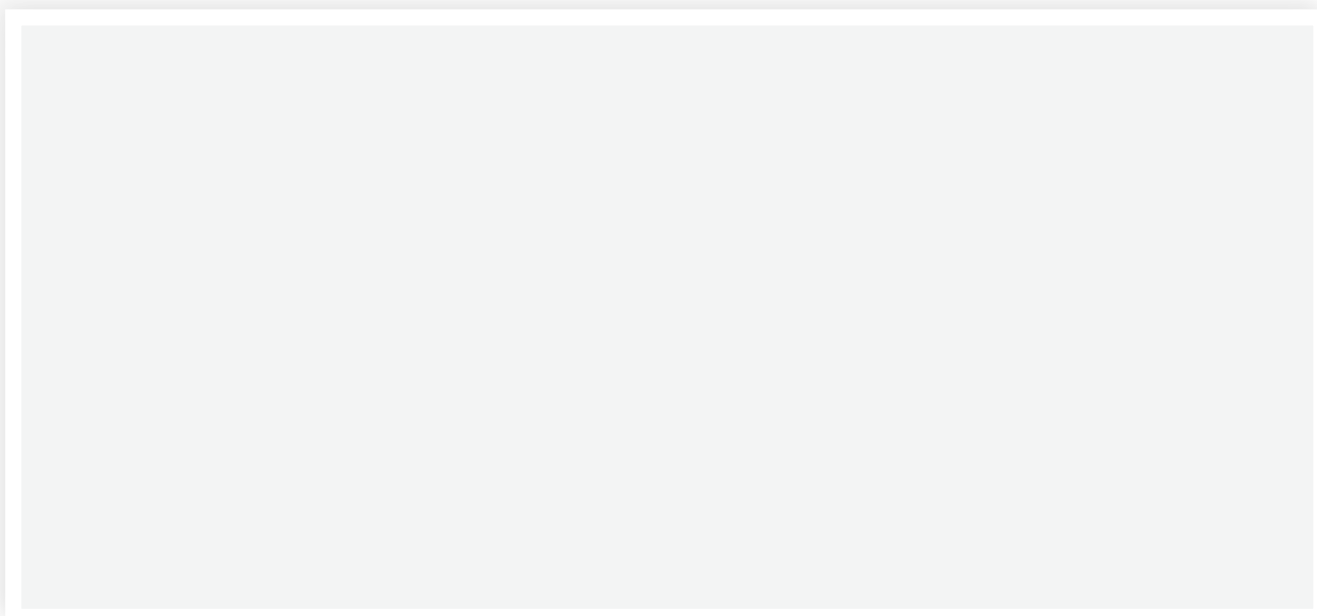
We are warned not to worry about technical debt: "we have to focus on new features"

We do some early warning: The engineers might bring them up, but mostly the early warning are ignored

We have them, but they need work to truly make an impact

Yes, we have that, and it works wonderfully

- Note any actions you should take to make improvements in this area.



Key 4: The Circle of Completeness (Feedback Loops)

How do runners know if they are getting better at racing? They have the data from races. They know their time for a 5K and if it is getting faster or slower. For any improvement activity, feedback loops are critical. They tell you if you are getting better. And improvement data provides motivation to do more.

Organizations that have defeated technical debt have feedback loops in place.

For example, for every technical debt item that engineers have fixed, they evaluate the actual results to the results estimated by the technical debt ROI calculator. They use these insights to make adjustments to their calculator and process. As these results are accumulated over time, they are used to reinforce the payoff with management, product owners, marketing, and other key stakeholders.

- How would you rate the effectiveness of your Circle of Completeness (feedback loops)?

- None exist

- What is behind does not matter. Forward, always forward, through the mud

- We have tried this, but we don't know how to get it right

- We are working on this, but need work to truly make an impact

- Yes, we have that, and it works wonderfully

- Note any actions you should take to make improvements in this area.

Key 5: **Perfect Balance: New Features Versus Speed Improvements**

If you have ever watched the Indianapolis 500 or similar car races, you will see a relentless focus on the getting a driver to the front of the pack. You will also see intense scrutiny of the effectiveness of the pit stops. Each pit stop serves to remove technical debt from the car! They replace tires that are wearing out and slowing the car down and many small adjustments to help the car go faster.

Organizations that have defeated technical debt achieve a perfect balance between customer-facing features and removing technical debt.

Most organizations have a roadmap (or at least a robust backlog) of desired customer features. That backlog or roadmap might include some technical debt items. But many organization struggle to make technical debt items a focus of their development strategy.

Organizations that have defeated technical debt have a perfectly balanced system for determining what items get worked on in each cycle. Technical debt maps, the tech debt ROI calculator, and feedback loops, are all important parts of that balance, but they are not the whole story. The decision-making groups have a whole **system** view for putting all those items together. They ensure the entire organization is getting faster over time and not being dragged back into the mud by technical debt.

■ How would you rate the effectiveness of your **perfectly balanced** system?

There is no balance, everything is focused on customer features: we have to sneak in tech debt fixes

Technical debt items get in there, but we must fight for them every time

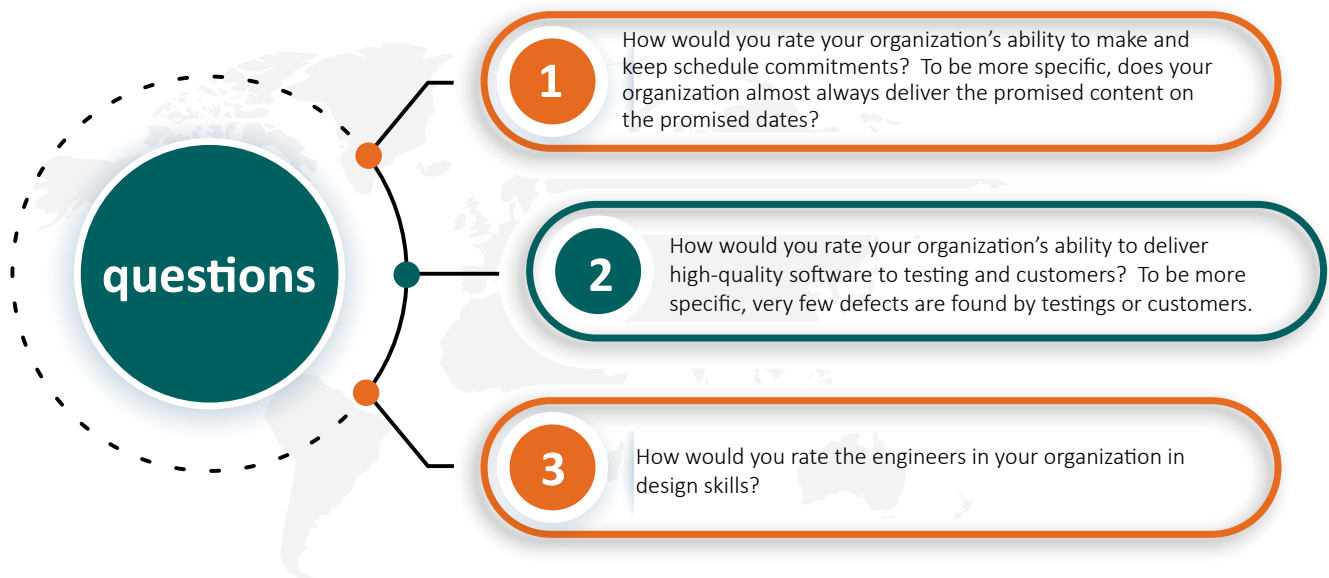
We are working on this, but need work to make a true impact

Yes, we have that, and it works wonderfully

■ Note any actions you should take to make improvements in this area.

One More Thing: These Keys Won't Exist Without a Foundation

Each of the five keys has a clear focus on technical debt. Yet it is hard for those keys to be built and sustained in an organization with some critical foundational elements. Consider these three questions.



If your organization is weak in any of these three areas, the Five Keys will help a lot, but there should also be work to ensure the foundation on which those keys are based is solid.

Note: If you want to have more information about the 5 keys and perhaps an example, or twenty, schedule a meeting with Exceptional Difference. We can provide some more context. Just email us at info@exceptionaldifference.com.

There is a Way Out!

Some people who take this survey find themselves discouraged at this point. Don't be! You've taken an important first step toward eliminating technical debt.

Maybe you've just realized that there is more technical debt in your organizations than you imagined.

Stop and take a breath.

We have worked with many organizations that were once plodding through knee-deep mud.

Some of the teams felt like they were going slower every production cycle – and when we put the measurements in place – we found out they were right.

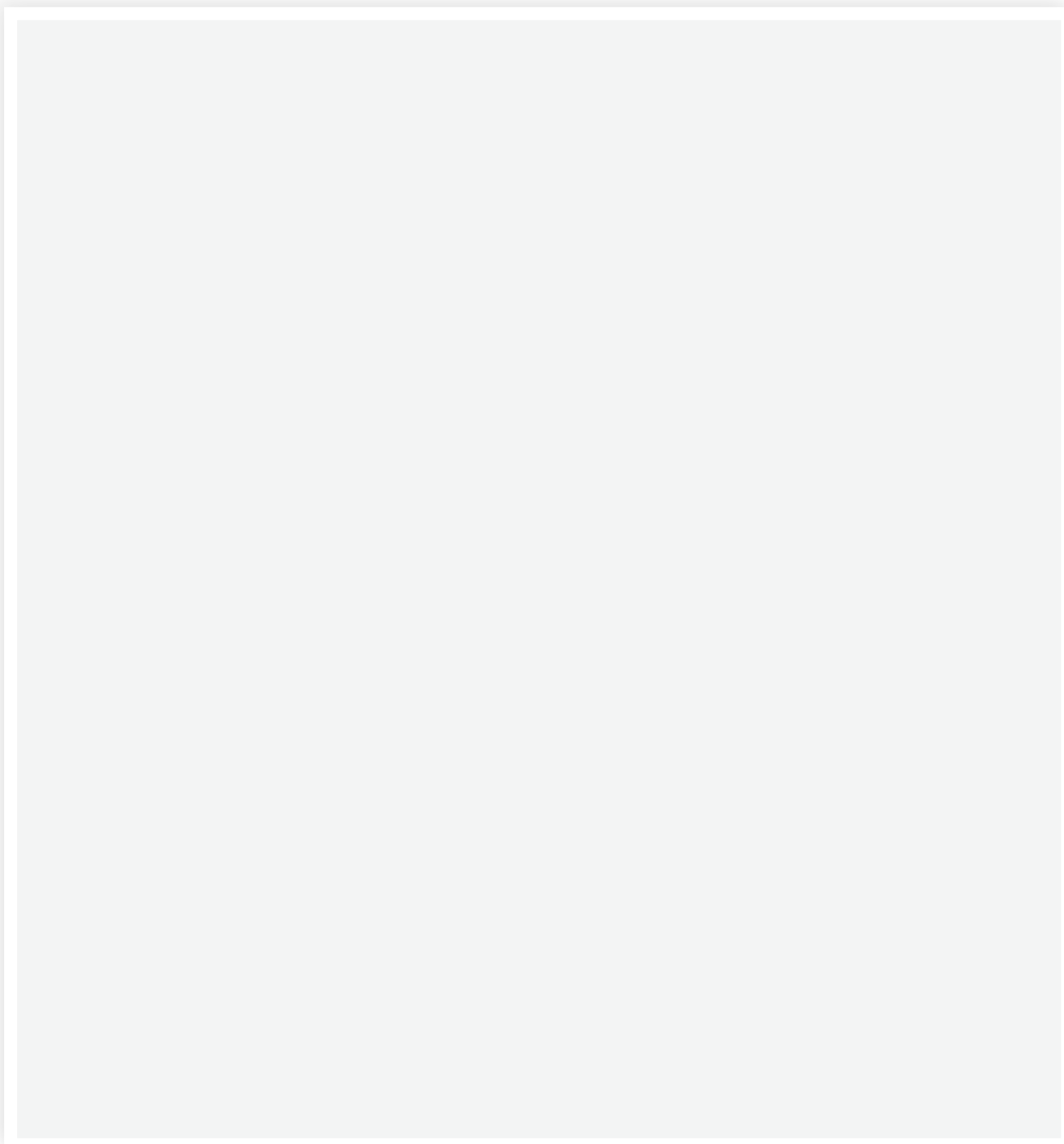
Know this: Many teams, even those in the deepest mud, have completely overcome their issues with technical debt. They put into place the five keys. They have achieved perfect balance. The engineers and leaders on these teams are now realizing their dream of building amazing products, unencumbered by technical debt.



No Assessment is Complete Without Taking Action

Getting out of the mud of technical debt calls for action.

What actions will you take right now to begin to address the issues you have identified?



"Each week I took what I learned and immediately applied it to myself and my team. We now all have so much more control over deadlines and quality." — **Francisco Javier**



For More Than Two Decades, We've Helped Engineers and Their Teams Gain Control of Their World and Build Better Software.

We know how frustrating it can be to build software when things just don't seem to click (pun intended).

We know how deflating it can be to put all of your ideas and effort into a product only to see it fraught with defect upon defect after a rushed product launch.

We know how challenging it can be to walk the tightrope between the development team and company management, massaging egos on both ends of the building. But we also know that great teams, great engineers, and great success stories are within reach of any organization.

It's the reason we created Exceptional Difference in the first place. We know there is a better way to deliver high quality software. One that doesn't take a toll on good people or sacrifice speed.

We've seen the real difference "a better way" can make in people's professional and personal lives.

After working with senior engineers and organizational leadership for 20+ years, we are still excited for each and every opportunity to help transform development teams into happier, more productive, and yes, healthier environments.

And to help elite engineers and team leaders reach their full potential and become even more exceptional in what they do each and every day.



Meet The Exceptional Difference Principals



Julia Mullaney CEO

With a commitment to changing the world of software engineering, Julia Mullaney, co-founder of Exceptional Difference, is an award-winning engineer, instructor, leader, and consultant who has worked with software companies in many industries across the globe.



Alan Willett Chief Engineer

Expert consultant, speaker, and award-winning author of *Leading the Unleashable: How to Manage Cynics, Divas, and Other Difficult People*, and *Lead With Speed*, Alan Willett is co-founder of Exceptional Difference. Alan works with clients around the world solving problems for organizations large and small.