

QualityClouds

Technical Debt Perspective

The State of Technical Debt
2023



Contents

How Does Technical Debt Impact Your Business	04
What is Technical Debt?	05
What Ways Is Technical Debt Introduced?	06
Why Do People Ignore Tech Debt?	07
How Might Technical Debt Impact Me?	08
Technical Debt Studies	09
How Can I Justify Spending On Technical Debt?	10
Technical Debt Doesn't Apply To Me	11
I Am Using A Low/No-Code Approach	12
How Can I Tell How Much Debt I Have In My Platform	1 3

Management Summary

Technical debt is one of the greatest risks organizations face in today's digital-first world and yet it is frequently ignored by businesses because it is hidden or misunderstood. Technical debt has been shown to reduce developer productivity and represents a significant risk to a business as it can impact the stability and maintainability of key systems.

Unmanaged technical debt is the cause of unexpected expense, reducing the return on investment (ROI), and in the worst cases can cause serious outages, and chronic performance issues with platforms such as ServiceNow or Salesforce the need to "re-platform" at a significant cost.

As businesses deal with the current financial squeeze, they are looking to technology to gain efficiencies. In this context, the need to "do more with less" can drive an increase in technical debt. So while lower-quality deliverables can meet the short-term need they almost certainly won't result in a cost reduction and will increase the risk of application issues down the road.

In this document we cover what technical debt is, how it is introduced, its impact, and techniques companies can use to manage and remove it from their platforms and applications.



How Does Technical Debt Impact Your Business

Technical debt is a term used in software development to describe the consequences of making trade-offs between short-term and long-term solutions. In software development, technical debt is incurred when a team chooses to use an expedient or shortcut solution to meet immediate needs, rather than taking the time to develop a more robust and scalable solution. Technical debt can arise in different areas of software development, such as architecture, design, coding, and documentation and testing. In the short term, taking on technical debt can help teams deliver software faster or meet a deadline. However, technical debt accumulates over time and can result in increased maintenance costs, decreased productivity, and decreased quality of the software. This can lead to stability issues, security vulnerabilities, poor productivity, and additional costs.

Technical debt is often seen as a metaphorical debt that accrues interest over time, similar to a financial debt. The longer technical debt remains unaddressed, the more expensive and time-consuming it becomes to fix. Technical debt can be managed through regular maintenance, refactoring, and by making a deliberate effort to reduce its accumulation.

It is important for software development teams to have processes and procedures in place to manage technical debt, even when it is introduced by external parties. This can include code reviews, testing, and documentation standards, as well as clear communication and collaboration between all parties involved in the software development project.

What Is Technical Debt?

There is ongoing debate among industry experts about whether tech debt should be shown on the balance sheet.

From an accounting perspective, tech debt is not considered a liability in the same way as financial debt, as it does not involve a contractual obligation to pay. However, some argue that tech debt should be recognized as a potential future cost and included in financial reporting to provide stakeholders with a better understanding of the long-term financial health of the organization.



One argument against showing tech debt on the balance sheet is that it can be difficult to accurately measure and quantify. Unlike financial debt, which has specific interest rates and repayment schedules, tech debt is more difficult to estimate and can depend on a range of factors, including the complexity of the codebase, the skill level of the development team, and the future direction of the business.

Ultimately, the decision to show tech debt on the balance sheet depends on the organization's accounting practices, as well as the preferences of its stakeholders. Some organizations may choose to disclose tech debt in other financial reports or disclosures, while others may prioritize other metrics in their financial reporting.

What Ways Is Technical Debt Introduced?

3rd Parties can represent a risk as they can introduce technical debt to a software development project. External parties, such as third-party vendors, contractors, or consultants, can introduce technical debt by providing code that is poorly written, undocumented, or non-compliant with the platform's best practice or with the companies' / projects' coding standards. Additionally, external parties may not be aware of the long-term implications of their solutions and may prioritize short-term goals over long-term maintainability and scalability.

In some cases, external parties may introduce technical debt unintentionally, such as when they are not familiar with the project's codebase or development processes. In other cases, they may introduce technical debt intentionally to cut corners, meet deadlines, or save costs.

Technical debt can also be added to the **platform or application via vendor changes.**

For example, if a vendor introduces a new feature that replaces an existing custom-built solution or an enhanced version of an existing feature, you may need to re-evaluate and modify your existing code to integrate with the new feature.

Alternatively, if a vendor updates their APIs or makes changes to their data model, you may need to modify your integrations to accommodate these changes. This is true for many organizations aligning to the ServiceNow CSDM. Failure to properly account for these changes can lead to "out of the box" applications not working, additional maintenance costs, lower performance, and a general increase in the risk of errors or failures.

To mitigate the risk of increased technical debt, it is important to stay informed of vendor updates and changes, carefully evaluate the impact of any changes on your existing customizations and integrations, and develop a plan to address any necessary modifications on time.

Why Do People Ignore Technical Debt?

- 1** Lack of resources: Companies may not have the resources to fully address tech debt. They may not have the budget to hire additional developers or allocate time for code refactoring and optimization.
- 2** Lack of awareness: Project stakeholders may not fully understand the risk relating to their technical debt, the potential consequences, such as system failure, extended outages, poor productivity as well as the overall impact on the long-term viability and maintainability of their software system.
- 3** Over-optimism: Developers may believe that they can address tech debt in the future or that it will not become a major issue. This can result in delays in addressing the problem, which can make it more difficult and costly to fix later.
- 4** Pressure to deliver: Developers may feel pressure to deliver quickly and may not have the time or resources to address tech debt. This can lead to a cycle of accumulating tech debt that is difficult to break.
- 5** Poor development coordination: Where businesses have multiple teams working on a common platform poor coordination between teams can lead to duplication of components, functions and the misalignment of meta data.
- 6** Short-term focus: In many cases, businesses are focused on delivering features and functionality as quickly as possible, and this can lead to shortcuts being taken in the development process.

How Might Technical Debt Impact Me?

Good hygiene relating to technical debt can significantly reduce risk, as well as help to avoid the significant costs associated with technical debt accumulation over time.

75%

Moderate to High Levels of
Technical Debt




According to a survey by Northbridge and Wikibon, technical debt is a growing concern for enterprise organizations, with 75% of respondents reporting that they have moderate to high level of technical debt in their applications.

In addition, technical debt can have a significant impact on end-users and business processes. A study by IBM found that technical debt can increase the likelihood of system crashes, downtime, and data errors, all of which can lead to lost productivity and revenue. The study also noted that addressing technical debt can improve software quality and reduce the likelihood of future issues, leading to long-term cost savings.

The importance of technical debt for platforms like ServiceNow and Salesforce cannot be overstated. The prevalence of technical debt in enterprise-level applications and the high cost of addressing it, highlights the need for developers and organizations to prioritize good software design and coding practices from the outset. By doing so, they can avoid accumulating technical debt and ensure that their applications are reliable, efficient, and scalable.

Technical Debt Studies


Good hygiene relating to technical debt can significantly reduce risk, as well as help to avoid the significant costs associated with technical debt accumulation over time.¹⁰



10%

Good hygiene relating to technical debt can significantly reduce risk, as well as help to avoid the significant costs associated with technical debt accumulation over time.¹⁰

- A survey by ManageEngine found that 61% of IT professionals said they spent more than 25% of their time dealing with tech debt.
- A report by Software Engineering Institute found that the cost of addressing tech debt increases over time, with costs rising by a factor of 3-5x if left unaddressed for several years.
- A study by Experian found that technical debt can negatively impact customer experiences, with 46% of consumers saying they would stop using a company's products or services after just one bad experience.
- A survey by AppDirect found that technical debt can hinder innovation, with 48% of respondents saying that technical debt limits their ability to create new products or services.



45%
Have Experience
Technical Debt
Issues

A report by Capgemini found that technical debt can lead to longer release cycles and lower quality software, with 45% of survey respondents saying they had experience these issues.

In summary, these examples illustrate the wide-ranging impacts of tech debt, from increased costs and decreased productivity to negative impacts on customer experiences and innovation. It's essential for organizations to proactively manage tech debt to avoid these consequences.

How Can I Justify Spending On Technical Debt?

Technical debt is not something that is isolated, it has a direct impact on many of the industry standard KPIs which are used to measure a business. Here are a few examples of organizations who prioritized technical debt in order to drive up customer satisfaction and their overall user experience.



British Airways experienced a system malfunction leaving 75,000 passengers at London's airports with flights canceled or delayed for days in 2017, due to technical debt. The airline industry was inundated with outdated systems and since then many airlines have made technical debt a priority to avoid future system failures.

Amazon, in 2018 announced that it had invested heavily in reducing technical debt in its cloud computing platform, AWS. This investment led to a significant improvement in the reliability and stability of the platform, which helped Amazon retain existing customers and win new business.



Microsoft, in 2014 embarked on a major effort to address technical debt in its Windows operating system. This effort led to a significant improvement in the stability and performance of the operating system.



Etsy announced that it had invested heavily in reducing technical debt in its online marketplace. This investment led to a significant improvement in site performance, which helped Etsy increase user engagement and drive more revenue.



In each of these examples, addressing technical debt led to tangible financial benefits, including increased customer satisfaction, improved reliability and stability, and increased revenue. These are all examples that show a direct correlation between the removal of tech debt and positive impact on outcomes.

Technical Debt Doesn't Apply To Me

Technical debt is invisible to a business user when it isn't causing an issue and if a system is already implemented and seen to be working there is a strong likelihood that it will be considered a "non- problem". This is because technical debt is typically a non-functional consideration, and it represents a risk and not generally an absolute certainty.

Functional components can be tested using traditional testing approaches, non-functional elements can be more tricky to identify and are defined by the environment being worked within.

Stability, Performance, scalability, manageability, system extensibility and other nonfunctional elements need to be considered.



These generally have contextual best practices associated with them which change as a platform or environment matures. With more people moving to cloud native systems which can be dramatically more complex, these non-functional considerations become more critical. This feedback frequently comes from smaller business users, who have implemented a SaaS application using a third party to configure or customize their implementation. They expect that the work is done and the system will run without the need for dedicated technical staff to do much more than basic application operations.

I Am Using A Low/No-Code Approach

Even though no or low code environments are designed to make application development more accessible to non-technical users, they can still be impacted by tech debt. Organizations using these environments should carefully consider the implications of customizations, code quality, integration, testing, and documentation to minimize the risk of technical debt. Here are some ways in which no or low code environments can be impacted by tech debt:

- 1** Code quality: Even though no or low code environments do not require coding skills, they still generate code in the background. If the quality of the generated code is not maintained, it can create technical debt.
- 2** Integration: No or low code environments often require integrations with other systems to extend their functionality. These integrations, if not done properly, can create technical debt.
- 3** Testing: Testing is critical to identify and fix issues before they cause problems in production. However, no or low code environments may not have a mature testing process, which can create technical debt by introducing issues into production environments.
- 4** Documentation: Proper documentation is critical to reducing the risk of future technical debt. However, no or low code environments may not have mature documentation processes, which can create technical debt by making it difficult to understand how the application is configured.
- 5** Customizations: No or low code environments often allow users to customize applications and workflows without writing code. However, if these customizations are not done following best practices, they can create technical debt.

How Can I Tell How Much Debt I Have In My Platform?

At Quality Clouds we provide a range of capabilities identifying technical debt in your current implementation as well as solutions for developers that prevent



- To identify tech debt our automated tooling starts by Identifying the areas of your application that have been changed frequently or have not been updated for a long time. These are the areas that are most likely to have accumulated technical debt.
- Look for common patterns in your codebase that indicate technical debt, such as duplicated code, hard-coded values, or long methods.
- Review your code to identify any architectural or design decisions that may have been made quickly without considering long-term implications.
- Use code Quality Clouds scan, to identify potential technical debt issues in your code. These tools can highlight areas of your code that may be difficult to maintain or that are vulnerable to bugs.
- Run a scan which will perform a review to identify any potential technical debt issues and prioritize them based on their impact on your application.
- Provide actionable insights for resolution and provide an estimate of the amount of effort required to address each item, allowing you to prioritize the issues based on their impact on your application and the effort required to fix them.

In Summary

Quality Clouds empowers businesses to maximize their SaaS investments and get the most out of their development resources. By combining visibility and automation in a comprehensive solution that gives you full control over your platform code, we ensure businesses can deliver a high- quality and sustainable experience to all your stakeholders.

Business success is increasingly dependent on the smooth running of computer systems and the applications that make up the services the businesses provide to their customers. The cost and complexity of building applications on a SaaS platform can slow innovation down and lead to increased risk. It's time to take a good hard look at technical debt in your environment so you can have a view on how optimizing your SaaS investment could reap both productivity benefits and make the application truly enterprise-ready.

As an independent authority on best coding practices, Quality Clouds will enable you to maximize your investment in your platform and get the most out of your development resources. Our solution offers granular and continuous visibility into your customized code, giving you an up-to- date and accurate picture of your existing technical debt. This ensures businesses have the data necessary to make informed decisions around debt and risk.

We also provide best practice validation using in-line code reviews, making sure that problematic code is never promoted to production. By enabling your developers to code better we free up time and resources to focus on delivering continuous value to the business. With Quality Clouds, you are assured that your code is of the highest quality, giving you peace of mind and confidence to innovate.

QualityClouds

20-22 Wenlock Rd / London N1 7GU
224 W 35th St, Suite 500 PMB 112 / New York, NY 10001
Carrer de Torres i Amat 21, 1^{er} / 08001 Barcelona
LONDON / NEW YORK/ BARCELONA



www.qualityclouds.com | sales@qualityclouds.com | [@QualityClouds](https://www.instagram.com/QualityClouds)