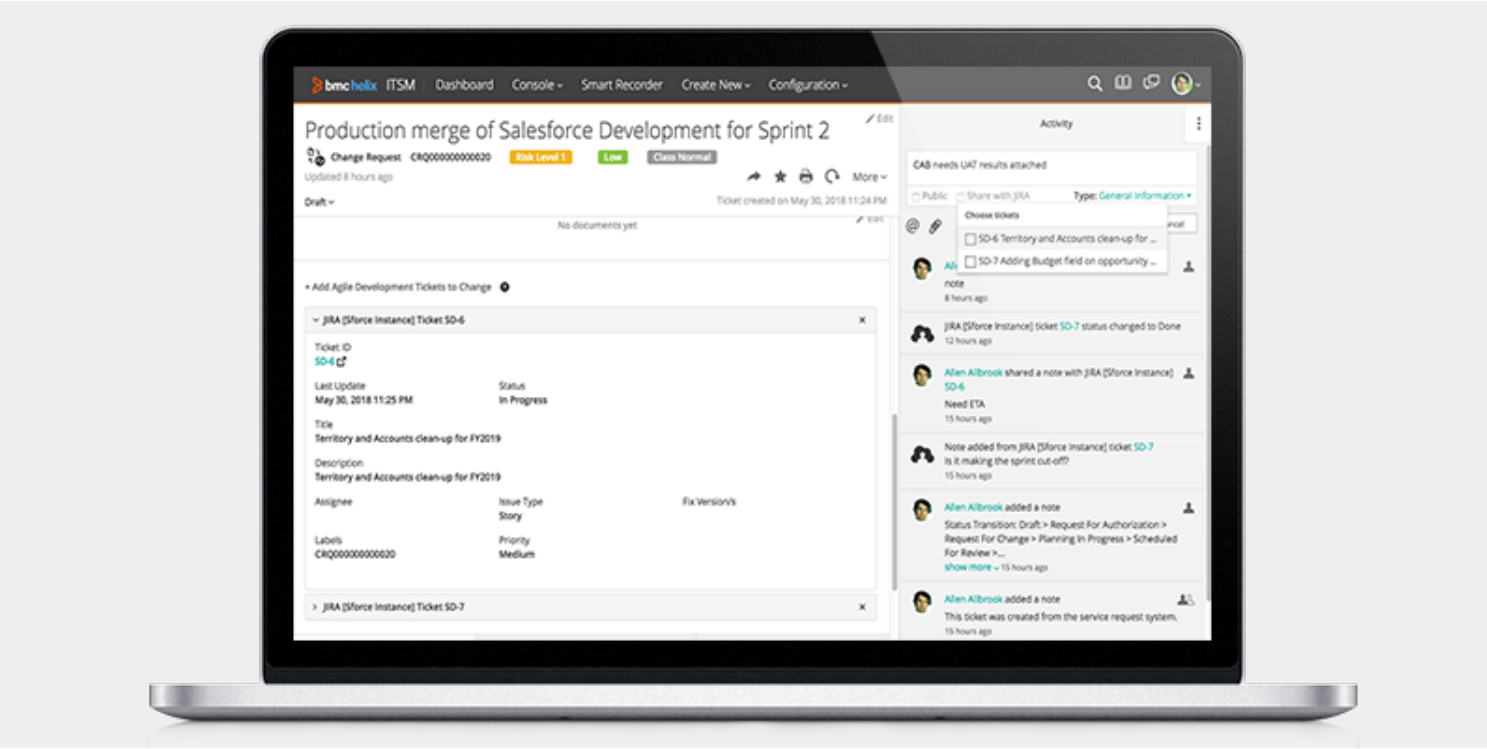
WHAT IS 'DARK DEBT'? HOW TO ELIMINATE THE DARK DEBT IN YOUR IT ORGANIZATION



How do you fix or prevent something from happening when you don't know it's a problem? That's the question inherent to dark debt – the hidden vulnerabilities in a stable IT environment that will eventually wreak havoc.

We've all heard of technical debt, a common IT concept, but dark debt is invisible. We don't know it's a problem until we see it. The way you approach this challenge is significant: ignoring dark debt will make your system failures worse, but discovering it, especially proactively, can help you minimize its impact on your IT environment.

What is dark debt?

Derived from the physics term "*dark matter*", which effects the world but isn't itself directly detected or seen, dark debt is a vulnerability in an IT system that is not known or visible until it causes a problem.

Dark debt refers to anything that happens that wasn't planned for or defended against. But dark debt is usually more than just a small mistake that wasn't caught in QC. Dark debt is often complex system failures, so complex that design logic ensures that a single fault in a system alone cannot create such a complex failure.

In any enterprise, IT handles vast and complex workloads that are dependent on other applications and systems. Importantly, software systems don't run just on code. At the most base level of any

system, software interacts with hardware, a whole other, elemental universe of silicon and electricity. These parts of the system are tenuously connected, and itis often impossible to detect or predict how subtle interactions between them will play out. As these interactions get more complex and spread across more factions of a framework, so, too, does the dark debt.

In both software and hardware development, we develop for functionality and defend against potential problems with a slew of backstops, like failovers, backups, exceptions and exception handles, and more. But the invisibility of dark debt makes it difficult to predict how or when these problems will rear their ugly heads.

Dark debt is not technical debt

Technical debt is the process of choosing a coding shortcut that can help speed things up so you can hit your product deliverables. Technical debt is a necessary evil, but it can be minimized by paying it off promptly. Three characteristics typically define technical debt:

- It's taken on wittingly and intentionally
- It's visible in code
- It can be corrected by refactoring – the disciplined technique of restructuring the internal structure of your code without changing its external behavior

Unlike technical debt, dark debt isn't limited to coding; it can occur anywhere and play out in infinite ways in any complex system. Importantly, dark debt is not a choice. It is beyond our control, until some anomaly in the system reveals the issue, often in a failure or unexpected outcome.

How to determine the source of a failure

Dark debt exists and it is invisible, so how can we possible protect against it? While you may never completely eliminate all dark debt, you can certainly minimize it. There are two paths to detecting dark debt: dealing with it defensively, wherein some failure or side effect reveals it, or dealing with it offensively, by actively seeking it out.

When a failure occurs, quickly identifying the source of the problem is essential. Your first inclination may be to swap a component at a time until you solve the problem, but that actually wastes more time.

Instead, diagnose the problem with data you have. Using a binary approach, identify two or more possible causes for your problem. Track down data that confirms or disproves each possible cause. Continue to narrow in on the cause until you have a reasonable sense of how to fix the weakness.

Of course, actively seeking dark debt may prevent a lot of late-night system failures. By testing a system's ability to tolerate failure while maintaining an acceptable quality of service, you can discover dark debt. This practice is called chaos engineering, and its goal is to build confidence around the system's ability to withstand unexpected situations or occurrences.

Pairing younger developers with senior developers ("*dark matter developers*") and opting for open-source code for resiliency and smarter design are other ways to uncover dark debt.

Reconsider your assumptions, too: we each have a "*mental map*" of how tools and systems interact, rightly or wrongly. We can't guarantee that these are correct until we are open-minded and test these assumptions, by checking logs and data reports for accuracy. Being open-minded is

especially important because our "*mental maps*" tend to remain static, when systems are anything but. Every change in a system reorients both dark debt and our understandings of how the system functions.

Once you've identified dark debt and taken steps to remedy it, consider writing up a quick post-mortem. Remember that the goal is not to determine what went wrong (you already did that), but instead to show where vulnerabilities lie.

One thing not to do? Firing the person you think caused it. Doing so is the perfect way to get rid of the one person who closely understands what went wrong.

Cultural dark debt

Dark debt doesn't have to hide in your hardware and software – it can fester in your organization's culture, too. Cultural dark debt can include any shortcuts that have a negative impact on your team. This type of dark debt is harmful in a "*soft skills*" way: if people can't feel that they can fail, they won't be able to learn or adapt in ways that benefit the company.

Indicators of cultural dark debt are any time failures result in blame and scapegoating, metrics encourage poor behavior, candor or frankness aren't welcome, and work is unplanned, late, or poor in quality.

Like dark debt in your IT systems, determining the source of the problem is essential. Net promoter score (NPS) surveys for employees can track whether your boss is open to negative news and how the team works, trusts, and learns from failure. By allowing for collaboration, debate and discussion are encouraged – which are the best ways to take a new idea and turn it into a strong idea.

In IT, change and failure are inevitable. The way you handle (or don't handle) both may constitute dark debt.