vFunction

REPORT

Conquering Software Complexity

Limitations of Conventional Approaches and Emerging New Solutions

Executive summary

Growing software complexity and technical debt are outpacing enterprises' ability to manage them, resulting in slow innovation velocity, delayed product launches, customer churn, and missed revenue opportunities, amongst other challenges.

vFunction conducted a qualitative study to dive deeper into challenges and trends uncovered in its May 2024 quantitative research report, "Microservices, Monoliths and the $1.52T Battle Against Technical Debt," which revealed factors that are leading to slower engineering velocity, limited application scalability, and resiliency issues. For this study, vFunction spoke with 12 CIOs/CTOs, software engineering leaders, and practitioners from large enterprises and growing digital-first companies to better understand challenges posed by increasingly complex applications and technical debt, and how they're addressing them.

The research shows that organizations manage diverse application portfolios, mixing custom and third-party applications, programming languages, and spanning on-premises, cloud, and hybrid infrastructures. These extensive software ecosystems complicate maintenance, updates, and innovation efforts for IT and development teams.

Enterprises recognize the need for advanced technologies and modern approaches for managing complexity and technical debt but have yet to adopt them. This report highlights limitations of conventional approaches and recommends emerging tools and processes to improve complexity and accelerate development.

12

CIOs/CTOs, software engineering leaders, engineers, and architects were interviewed

Average custom applications

Digital-first companies

~3–50

Large enterprises

~100–400+

A majority have a mix of monoliths and microservices

Several are in the process of transitioning from on-premise to cloud or hybrid environments

Most use a range of programming languages, from legacy COBOL to modern Go and Ruby, supported by various development cultures

The growing complexity challenge

Managing complexity, optimizing development, and reducing technical debt has become increasingly challenging. As one software engineering manager at a multinational location and consumer electronics tech company noted, "It's a lot. Almost one-third of our team capacity is going toward managing tech debt."

Faster releases, emerging tech integration, and microservices adoption have complicated application environments. When vFunction asked 1,000 U.S.-based architecture, development, and engineering leaders about the state of their application development, it found that nearly half of organizations are grappling with:

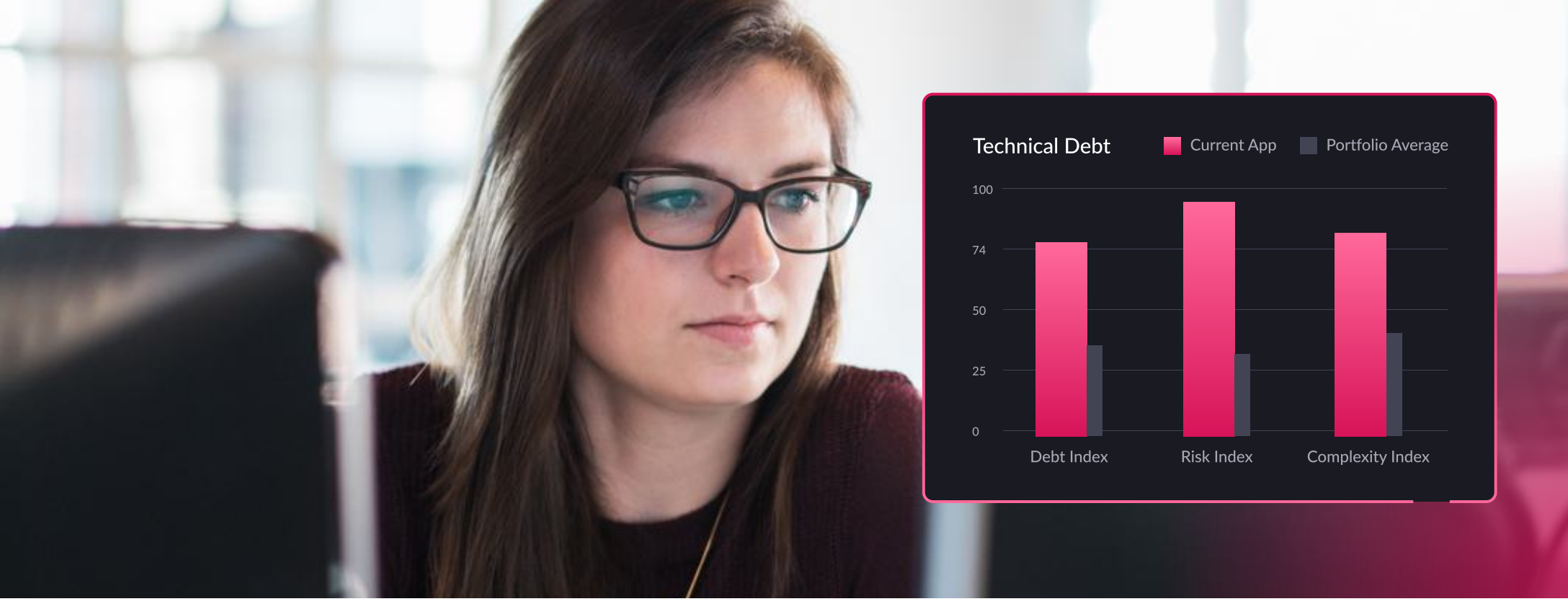
48%	42%	42%
Extremely slow or moderate engineering velocity	Limited, poor or moderate application scalability	Extremely low or moderate resiliency

In conversations with CIOs/CTOs, software engineering leaders and software architects, several key factors emerged as primary contributors to software complexity:

- Maintenance of legacy systems
- Integration challenges following acquisitions
- Budget and skilled personnel constraints
- Pressure for rapid solutions amid business changes
- Skills gaps and knowledge transfer issues with retiring and departing developers
- Lack of standardization across tools, processes, and code, too often relying on manual processes and familiar solutions

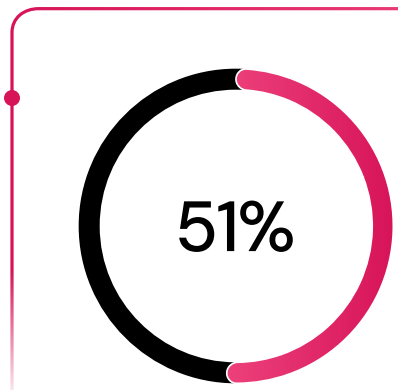
These factors collectively impact business operations, leading to slower feature delivery, decreased engineering velocity, and difficulties in system maintenance and updates.

> "It's particularly challenging when we have developers who are accustomed to one language and development philosophy trying to migrate to a new language. For example, we looked at moving to Go, and some developers tried to replicate the Ruby development environment in Go for speed benefits. This approach cost us dearly because the philosophies don't translate well....the resulting technical debt was quite drastic.
>
> **- Senior software engineer at a leading platform of tools for creators**



The hidden costs of technical debt

According to both research studies, budgeting for technical debt remediation often competes with new feature development. vFunction's quantitative study found:



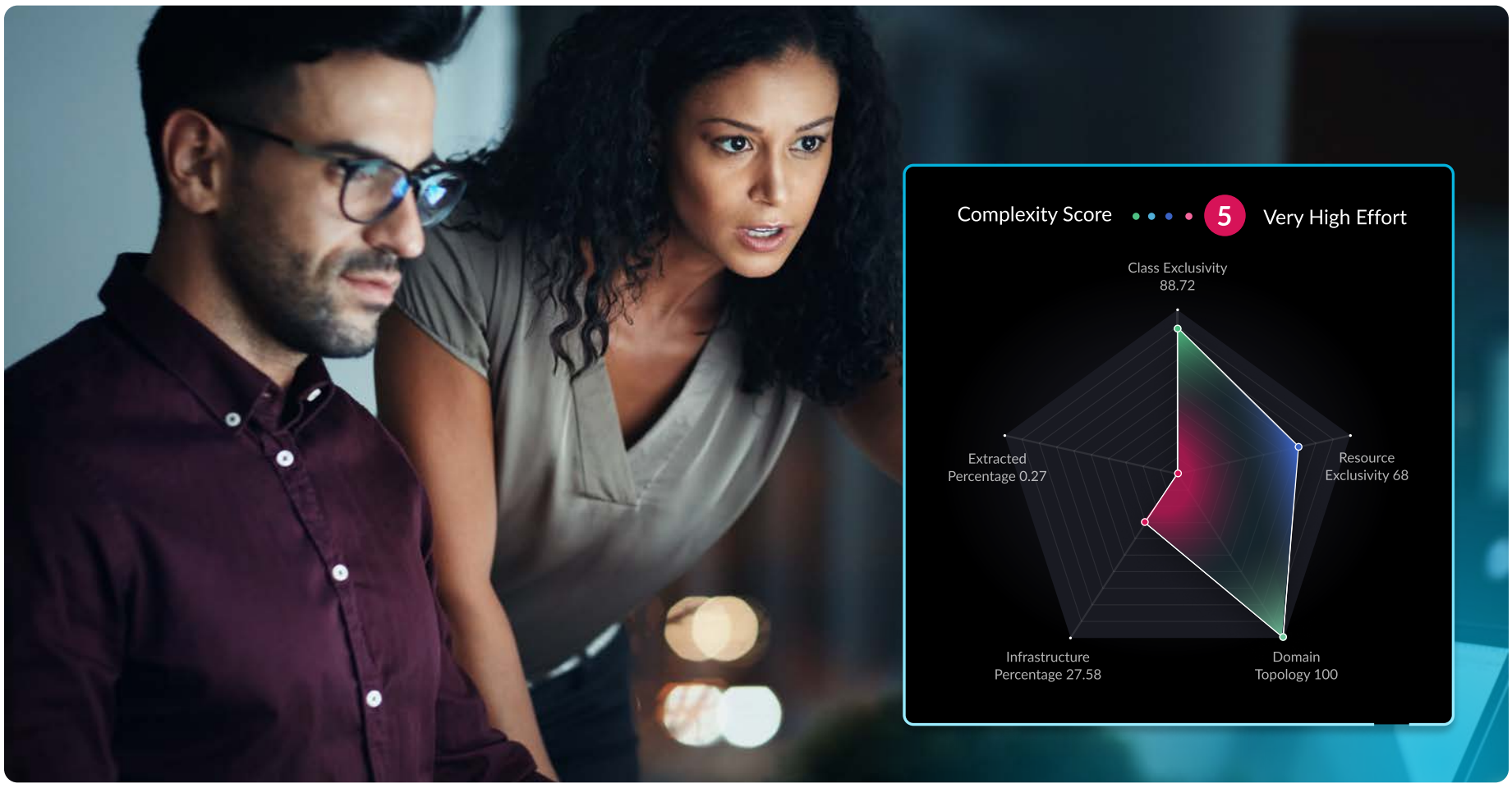
51% of organizations spend **over 25%** of their total IT/engineering budget on technical debt remediation

The qualitative study corroborated substantial resource allocation to managing technical debt. Most participants reported dedicating 20-30% of their IT budget to remediation efforts, with some exceeding 50% of the IT budget.

> "I would say far more than 50% of our work has been addressing the backlog of the past 25 years...70-80% is taking care of the technical debts of the past..."
>
> - CIO and CTO of leading provider of industrial tools and equipment

Long-term consequences of complex applications and unaddressed technical debt can be severe and slow enterprise innovation.

> "It is tough because we understand where we want to be with a new paradigm. As an organization, we want to be more in the microservices approach versus the monolith approach. But again, revenue is a big driver for our organization. So it's tough to do what is essentially non-revenue-recognizing work, meaning cleaning up that technical debt, breaking up the monolith into more microservices, and centralizing our data stores."
>
> - Software architect at a cloud-based human resources and recruiting software company



Ad-hoc, manual approaches to complexity and technical debt stifle innovation

Organizations vary widely in measuring and addressing complexity and technical debt. Many lack standardized assessments, consistent prioritization, or dedicated remediation resources. Some use architecture guilds or development excellence teams, while others implement a combination of team member interviews and tools. However, these approaches are often manual and lack application ecosystem-wide visibility.

"From an architectural standpoint, understanding complexity typically involves a lot of interviewing people about the components and how they work together.

People verbally describe what the interfaces look like, and maybe they have a spec, but it's probably out of date. Then you try to put that picture together."

- Sr. principal software architect at an international data management company

Aligning leaders and teams

Complexity and technical debt management often lack clear ownership.

CIO/CTO perspectives

According to CIOs/CTOs interviewed, half take responsibility (along with their direct reports) for ensuring routine system management before complexity and technical debt reach a boiling point. The other half rely on other teams and resources.

"IT is responsible...I'm leading the team, but it's not me personally...it's delegated to all the application and platform owners and architects."

- CIO and CTO of the leading provider of industrial tools and equipment

"We don't actually have that individual that looks across all of our apps from a technical coding perspective. We've assigned two lead developers to a set of apps...we need someone in more of a head role again."

- Group CIO of a healthcare consultancy and advisory firm

"My VP of infrastructure, CISO, VP of App Dev, VP of Data and I contribute to planning and execution. Then we present to the CEO/C-suite... Consultants help implement it and make architectural recommendations, working with my IT team to come up with a plan."

- CIO of a cash management company

Addressing technical debt requires a strategic vision, clear ownership, organizational influence, and product authority. Without these, feature churn and fragmentation persists at the expense of meaningful improvements. While quick fixes offer temporary relief, unchecked technical debt will eventually manifest in more outages and vulnerabilities. Proactive, continuous management of technical debt is essential throughout the software development life cycle.

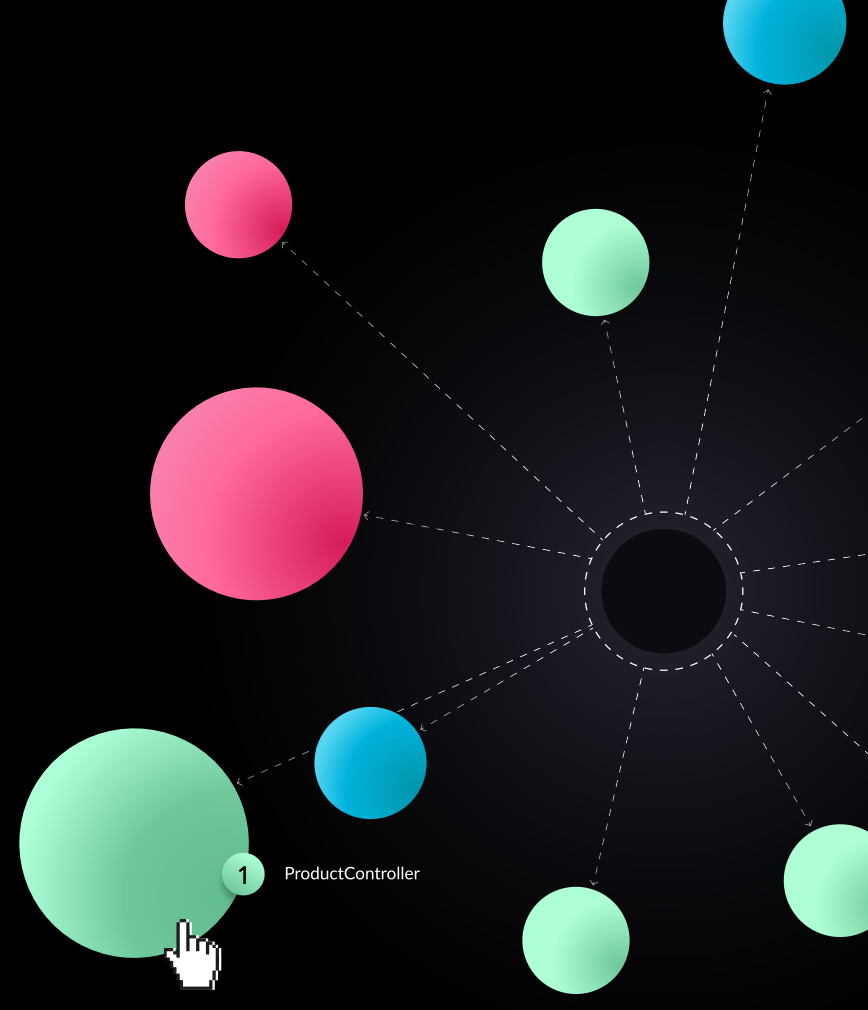
> "Some people say, 'We'll see when we get there.' But that's not the right approach. I think that mindset comes from product teams saying, 'Let's deliver this and we'll see when we get there.' That's what I'm trying to change in the organization. We should not think short-term but start thinking long-term in terms of how we build, architect, and deliver. That's the kind of view I think most of us should have."
>
> **- Senior data engineering manager at a major commercial bank**

The absence of standardized approaches and clear ownership in managing software complexity and technical debt highlights the urgent need for systematic, widely-adopted strategies that balance innovation with streamlining application ecosystems.

Emerging solutions

The case for architectural observability

Architectural observability is emerging as a critical capability. It uses AI to analyze applications statically and dynamically to provide a comprehensive understanding of software architecture, enabling teams to measure and prioritize technical debt and reduce application complexity with every release. It helps improve software modularity, identify the sources of technical debt, enable architecture governance, and facilitate modernization efforts for more scalable and resilient applications.



There is a shared consensus among CIOs/CTOs, software engineering leaders, and practitioners on the need for architectural observability and the ability to observe and understand software architecture in real time.

As the deputy CIO at a multinational investment bank and financial services company said:

> "**For the end developers** doing the actual coding, it shows them what's never being used, what's outdated, and what hasn't been touched."

> "**From a development manager's perspective**, who's under pressure to get these applications updated and up to par... It provides clear insights into the state of the codebase, allowing for more informed decision-making and prioritization."

> "**For the infrastructure team**, it's invaluable to understand where legacy code exists or where code is running on legacy infrastructure."

Embracing AI and automation in development

The conversations also revealed significant optimism about AI-augmented development and for conquering complexity:

- Improving CI/CD pipelines
- Automating code development and testing
- Enhancing security controls
- Facilitating integration planning

GenAI tools have the potential to increase development velocity, improve code quality, and help teams quickly adapt to changing business needs and technical debt challenges.

Alongside these tools, organizations want consistent methods for assessing and remediating technical debt, including clearer ownership and accountability, and better integration into development cycles.

These advanced tools and processes offer a systematic, data-driven approach to complexity and technical debt management. The ultimate goal is to strike a balance between driving innovation and maintaining healthy, sustainable application ecosystems that can adapt to changing business needs.

vFunction: Empowering enterprises to tackle growing application complexity

Solutions providing architectural insights and actionable recommendations are crucial for maintaining agile software development and preventing technical debt accumulation.

vFunction uses AI-driven architectural observability to analyze application architectural health, providing deep insights into application structures and recommendations for improvements. It maps dependencies, functional domains, and entry points across portfolios, and assesses domain exclusivity and component interactions to determine modularity and identify critical areas of technical debt.

To learn more about how vFunction can help your organization tackle software complexity and technical debt, request a demo today.

Request a demo

Backed by

Partnered with

Awards & Recognition

About vFunction

vFunction, the pioneer of AI-driven architectural observability, delivers a platform that increases application resiliency, scalability, and engineering velocity by continuously identifying and recommending ways to reduce technical debt and complexity in applications. Global system integrators and top cloud providers partner with vFunction to assist leading companies like Intesa Sanpaolo and Trend Micro in discovering their architecture and transforming applications to innovate faster and change their business trajectory. vFunction is headquartered in Menlo Park, CA, with offices in Israel, London, and Austin, TX.

To learn more, visit www.vfunction.com.