Solving Software's R&D Conundrum

OCTOBER 05, 2022

By Pranay Ahlawat, Clark O'Niell, Archith Mohan, and Ted Wiles

> **Smaller software companies tend to outperform their larger competitors in returns from R&D investments. How can software companies ensure the scaling of R&D even as they grow?**

Software companies live by the mantra of "innovate or die." As a result, research and development programs are among their most indispensable functions. The average software company spends about 20% of revenue on R&D; some spend more than 40%. The effectiveness of R&D teams—chiefly, their ability to innovate—is in many cases a software company's most significant differentiator.

Unlike most other aspects of business, however, R&D doesn't scale with size. Indeed, recent BCG research shows that smaller companies tend to outperform their larger competitors in returns from R&D investments and speed to market. This may seem obvious—after all, older companies generally have more mature products in slower-growth markets—but we found that market maturity alone doesn't explain declining RDIs.

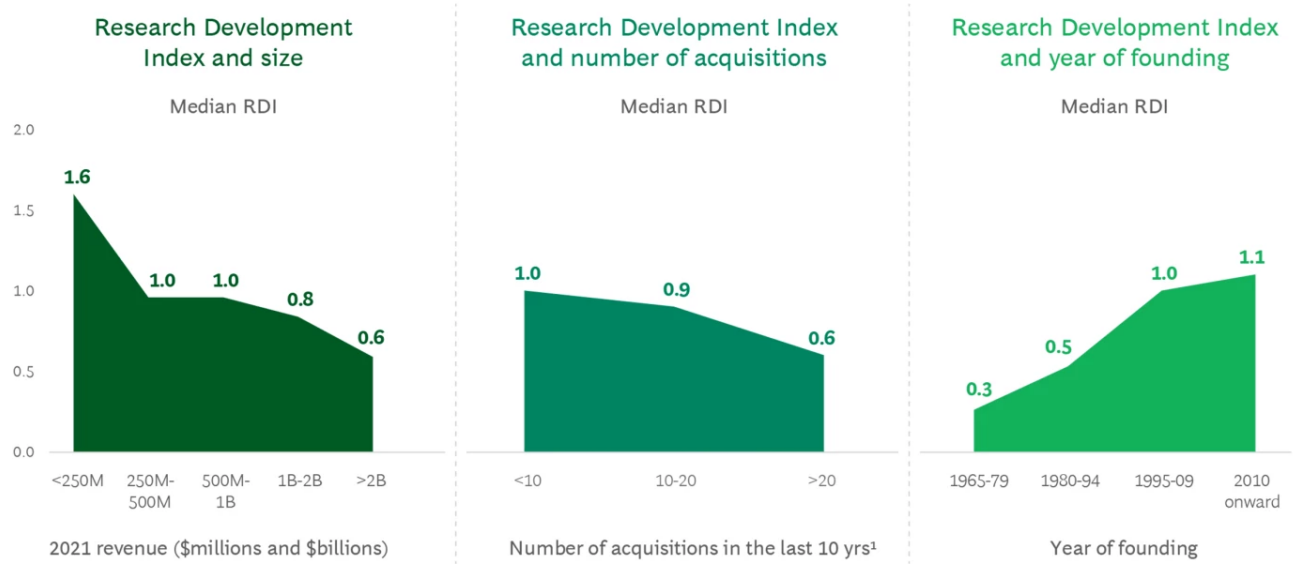
> "Unlike most other aspects of business, R&D doesn't scale with size.

In fact, the nuances and outliers in our analysis convinced us that we had to look deeper and answer these questions: What separates the software companies that have successfully scaled their R&D from the many competitors who have not? What are the factors that get in the way of companies as they scale? Finally, what can software companies who want to grow without losing their edge in innovation do to solve the R&D conundrum?

When R&D Lags

BCG recently analyzed nearly 200 mostly public software companies from around the globe to compare the change in one-year revenue growth (excluding acquisitions) against the percentage of revenue allotted to R&D.[1] This ratio is what we call the Research and Development Index (RDI). The companies we examined are young to old, small to large, and with varied growth strategies. Our analysis shows that as a company matures and increases in size, its RDI falls, in large part because its R&D budget expands commensurately while not delivering an equal amount of revenue gains. (See Exhibit 1.)

Exhibit 1 - R&D Performance Decreases as Companies Become More Complex



Source: BCG research.

Note: RDI = Research Development Index. Simple RDI divides the change in organic revenue (which excludes acquisition revenue) from year 1 to year 2 by R&D spending as a percentage of revenue in year 1. For RDI calculations here, revenue growth in FY21 over FY20 is divided by FY20 R&D expense as a percentage of revenue. BCG analyzed nearly 200 mostly public software companies from around the globe.

¹January 2012–January 2022.

By contrast, smaller companies, under $250 million, which tend to focus on product development and incremental product improvements, have median RDIs as high as 1.6, boosted by rapidly compounding growth rates. But as software companies surpass $2 billion in revenue, their RDI tends to decline precipitously (median values of 0.6), often the result of fewer new product features and advances.

RDI also tends to be worse in companies whose growth is based on acquiring other companies. Few acquisitions achieve the hoped-for product and operational synergies, often because the two companies have separate technology stacks that are difficult to integrate. Consequently, the more acquisitions that a software company makes—which usually means that they are deferring organic growth—the more its RDI suffers.

Some software companies defy this pattern, but it is hard to place them in precise categories. On the one end of the spectrum of the outliers, there are some larger software outfits with more traditional enterprise sales business models—such as Adobe, Microsoft, and Salesforce—that maintain decent RDIs. On the other end of the spectrum, a few

product-led companies, whose growth strategies are driven by attracting customers to a flagship product line, have low RDIs. These outliers only make the R&D conundrum more perplexing.

Why R&D Efficiency Lags

Broadly, we attribute the inability to scale R&D to increased organizational complexity, a natural but problematic by-product of growth and maturity that companies often fail to create strategies to control. The types of complexity that hinder efforts to scale R&D can be broken down into four categories. (See Exhibit 2.)

Exhibit 2 - Types of Complexity That Weigh Down R&D Performance

Source: BCG analysis.

Let's examine each of these categories in greater detail:

Engineering and Technology. Older, larger companies are often wedded to aging products and technologies. Frequently, this happens because their legacy products represent their largest revenue streams, and there is little organizational will to risk losing the sales tied to their core product platforms and architectures. These products have often accumulated technical debt, such as leftover and obsolete code, bugs, and design requirements that are a drag on continuous innovation and modernization.

Companies stuck in this situation face increasing technological irrelevance as new application development frameworks and platforms—such as cloud and serverless computing, next-generation data storage, and AI—shift the economics and speed-to-market of product development. These advances, which many newer competitors have eagerly embraced, have made it easier for new entrants to race past older companies in specific markets with essentially turnkey products and to maintain their lead through rapid improvements. Consequently, R&D outlays at older companies for less relevant products decrease in efficacy and value.

We have witnessed this play out at numerous companies, including a leader in gaming and operations center software that has been beset by engineering and development slowdowns and redundancy due to its large and complex code base, which grew unchecked over many years. In addition, multiple programming languages, among them legacy versions of C/C++, and an inefficient data model have created a shaky foundation for innovation. Recognizing the problem and seeking a return to growth, the company has begun an effort to systematically modernize and re-write its products.

Organization and Operating Models. Many software companies serve as apt exemplars of Conway's law, which states that a company's software design function and software architecture reflect the organization's ability to collaborate and communicate. Said another way, building scalable products requires thoughtfully orchestrated operating models. Many larger, multiproduct companies have overly complicated development structures because they have become bloated, with a raft of siloed teams working on different parts of the same products. Collaboration and coordination across product lines, features and functions are hindered while teams become larger and more stratified. And the development work itself may be redundant and duplicated across teams. In this insular environment, planning, coordination, and product development—and, as a result, R&D efficiency—suffer.

Sometimes operating model complexity comes from inorganic moves—primarily acquisitions—to expand growth. To fully integrate acquired businesses and products companies have to combine multiple operating models with diverse technology stacks and varied planning and software development teams. In many cases, companies give up

trying to combine separate offerings into a logical product portfolio and instead allow them to continue their disparate efforts with little or no integration. This creates duplicative development functions and operations and sometimes redundant portfolios and, in turn, customer confusion. As a result, it is virtually impossible to drive the intended synergies and economies of scale.

In addition to its impact on productivity, acquiring companies without a fully thought-out integration plan can lead to downstream problems in workforce retention and further erosion of company performance. One software company we studied took a laissez-faire approach to a significant acquisition by combining teams of engineers into a single pool without a clear roadmap for their roles and their product development priorities. The result: more than half of their best talent left the firm, frustrated by the opaqueness of the company's integration tactics and expectations.

Relationships with Customers. There are two types of customer dynamics that hold software companies back. First, younger software companies or ones operating in markets with large customer concentration are overly reactive to their so-called highest annual recurring revenue (ARR) customers in designing new product features that these ostensibly valuable customers demand but that few other customers will pay for. This impacts R&D efficiency, which depends on building products that scale to large markets and multiple customers; by contrast, prioritizing bespoke, or customized, features in the near term inevitably hurts product economics in the medium to long term. In other cases —particularly those involving older companies—customers are responsible for stymieing attempts to upgrade products without adding operational complexity. Software companies with legacy on-premises products are particularly susceptible to this problem and face the challenge of supporting multiple versions of their products, greatly increasing R&D and support burdens. At the same time, many companies are loath to force these long-standing customers to modernize by embracing SaaS- or PaaS-cloud-based or turnkey products. Even if they are upgrading and innovating their products, the inability to turn their backs on older customers forces them to adopt two-speed models—developing, selling, and supporting different modalities of the same or similar products. These somewhat dysfunctional relationships with customers place software firms in a quagmire, held

hostage by their older—and often largest—customers and hurting their ability to modernize and compete.

> Missing market windows by a few quarters can mean the difference between building the next billion-dollar product or losing out to competitors.

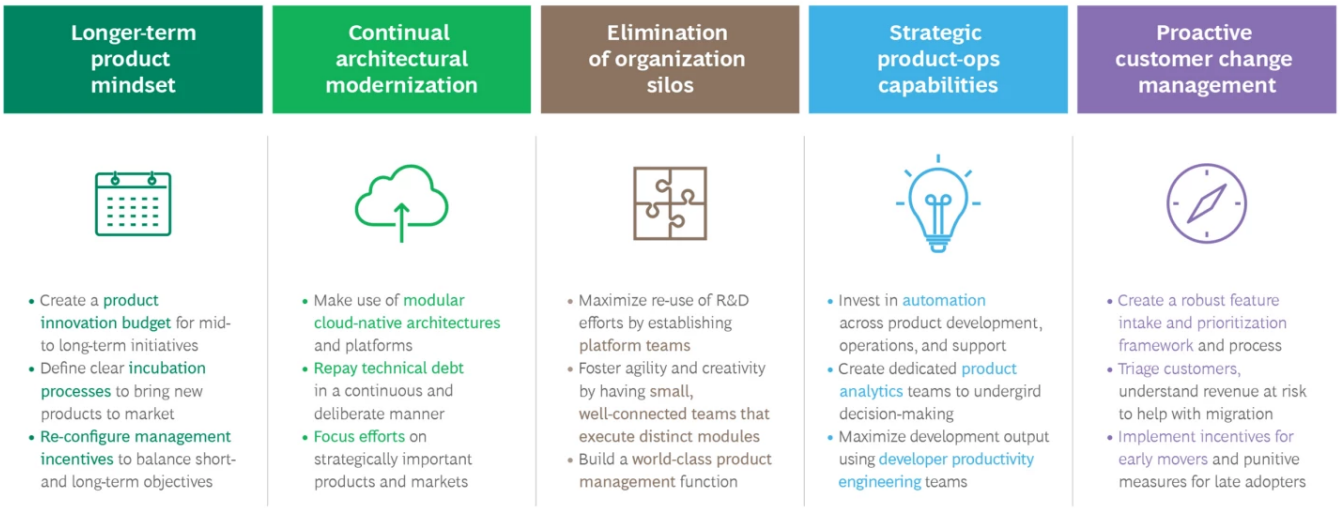
Short-Termism. For many incumbent software companies, short-termism is, unfortunately, the way of doing business. It is the manifestation of the Innovator's Dilemma, formulated by Harvard Business School professor and BCG alumnus Clayton Christensen. The Innovator's Dilemma holds that boards and executives of established companies fail to value new innovation sufficiently because it often doesn't bring the immediate returns that can wrung out of existing customers and product architectures. In other words, company leaders reject going to market with new, strategic, or adjacent products because these products' short-term potential appears too limited to drive higher revenues and their perceived risk too high. In the world of technology, where trends change rapidly and success depends on staying ahead of the curve, missing market windows by a mere few quarters can often mean the difference between building the next billion-dollar product or losing out to competitors. We have seen this play out with cloud-based products, data platforms, and applications software across all categories.

Short-termism is, of course, a myopic attitude that deters effective planning for the future. But it is nonetheless adopted by many CEOs and other executives whose incentives and jobs are linked to next quarter improvements as opposed to longer-term value creation. In the end, this approach leads to misallocated portfolios and stands in the way of companies innovating, maintaining competitive products, or making the right types of investments to scale R&D and sustain long-term growth.

How to Scale R&D

Complexity can be daunting, but we have identified a series of specific steps that software companies can take to avoid its worst aspects. In the process, companies can create pathways for scaling research and development, allowing revenue gains to outpace R&D earmarks. (See Exhibit 3.)

Exhibit 3 - Steps to Scale R&D and Product Development



Source: BCG analysis.

Adopt a long-term product investment strategy while maintaining an effective portfolio in the short term. Difficult though it may be, striking the balance between the short term and the long term is essential for ensuring that R&D and product development are aligned with growth strategies and value creation. Companies need both an effective portfolio that consistently delivers strong quarterly results and to undertake innovation planning for the future.

A long-term product investment plan should ideally include three elements:

First, the plan must have a clearly articulated innovation budget for medium- and long-term product development initiatives. This budget should be flexible enough to shift

priorities as new technology or market expediencies arise. And it should be disciplined: no more than 10% of total R&D earmarked for longer-term moonshots.

Second, the plan should include a defined process to incubate and bring new products to market and to do so without bureaucratic roadblocks or technological inertia. Allocating R&D capacity alone is not enough. Companies should adopt a software lifecycle process and framework that prioritizes new products, measures their success, and manages them differently from legacy or cash-cow offerings.

Third, the plan must feature incentives that minimize short-termism. Managers should be rewarded for supporting longer-term projects that ultimately drive greater growth. Bonuses can be based on revenues from innovation portfolios, a number of new products that surpass a certain sales threshold over a period of time, or market expansion, among other possibilities.

However, long-term innovation gains are only possible if product portfolios are well managed in the short term. That means companies must avoid placing too many bets on too many products and must be willing to shelve products that are not successful. They must avoid technologically cumbersome and fragmented portfolios in favor of simple product lines with two or three anchors that can be counted on for consistent revenue streams. With this approach, R&D is focused on designing adjacent applications and improving features and functions in a limited but expanding universe, resulting in targeted growth channels, software development efficiency, and a more efficient use of resources.

Top-flight software companies from Apple to Microsoft have all adopted this reduced portfolio model. Indeed, to paraphrase Steve Jobs commenting on the dangers of product proliferation, the hallmark of a good company is not always saying yes, but rather saying no to 1,000 other good ideas that would only serve to distract.

Track and repay technical debt to drive continual architecture modernization. Software architecture is always evolving. Our analysis found that software companies with the highest RDI offer their products on the most-modern software platforms. In today's technology landscape that means having a highly modular and cloud-native architecture

and designing software using modern automated applications development and deployment methodologies. Moving to such scaled architectures accelerates innovation, improves product acceptance, and makes it easier to update products frequently.

Established companies with older technology stacks should build a modernization roadmap to better compete with newer rivals and enjoy the benefits that hyperplexed architectures offer. Best-in-class software companies earmark roughly a quarter of their product development budgets for platform modernization and automation and repaying technical debt in a continuous and deliberate way, our research found.

> “
>
> Avoid massive re-architectures of aging technology stacks related to products that are past their expiration date.

However, it is important to target modernization efforts on faster growth or strategically important products and markets, an imperative that many companies make the mistake of neglecting. Massive re-architectures of aging technology stacks related to products in markets that are past their expiration date is usually a wasteful exercise, a distraction from more profitable investments that better align with a company’s growth plans.

Break organization silos by investing in shared services and product management. To overcome the organizational complexity and silos that are perhaps the biggest impediments to scaling R&D and product development, these steps can prove valuable:

First, maximize re-use and sharing of R&D efforts. In an ideal structure, a software company should have a shared services or a platform team that builds common data and middleware components, user interfaces, security services, and tooling frameworks for all products. This team’s primary strategic roles are to ensure code re-usability, drive faster time to market for application features, and improve the overall user experience by making the software more integrated and consistent with the company’s portfolio of

products. One large SaaS fintech that adopted this shared platform approach found that the more efficient product development model and improved interoperability generated significantly higher returns from cross-selling.

Second, individual product development teams should be small, self-sufficient and, at the same time, collaborative and accountable for results. As much as possible, work should be executed in a modular and self-contained way with each group focused on specific product development or upgrading tasks with clearly defined interactions among development teams. This helps ensure that progress is shared, and new ideas proliferate. Amazon embraced this concept with its “two pizza team” approach, in which units of work are executed and owned by a team of six to eight engineers.

Lastly, companies must invest in world-class product and portfolio management capabilities. Possibly the single most important but often overlooked aspect of increasing returns from R&D, product and portfolio leadership not only bridges product teams with each other but also connects these teams to other functions, such as product marketing, sales, and customer support. Product management leaders help product development teams focus, prioritize tasks and measure performance and results. Indeed, skilled product development and management leadership is the single biggest determinant in helping large product teams scale effectively.

> “How to scale large product teams effectively? Skilled development and management leadership.

Invest in product scale multipliers. To guard against complexity creep, companies need to be proactive by adopting what we refer to as scale multipliers. These are essentially capabilities solely designed to continuously maintain scale improvements and ferret out potential problems that would hinder them. Scale multipliers come in many forms but in our view early adoption of automation, product analytics and developer productivity engineering (DPE) is a good starting point.

Companies tend to be reactive when implementing automation, adding it to product development late or only when a large-scale inefficiency is discovered. But when automation is embedded in the entire product development process—across functions from quality and testing to release engineering, security, and operations—scaling becomes ubiquitous. This means positive impacts for product quality, performance, customer satisfaction, and retention.

Similarly, product analytics is usually an after-thought at many enterprise software companies. Companies decide on new product features and even new products, and sometimes make poorly thought-out investments in incremental improvements, without fully understanding their customers' preferences and usage patterns. Investing in data capture technologies and building a dedicated product analytics team helps overcome this, supporting the development of products that best address customer needs and that improve the customer experience while undergirding data-driven product and portfolio management across the organization.

DPE targets complexity seeping into the code base and product development tooling and processes that tends to occur when engineering companies' portfolios expand, and new features are added on more frequently. DPE confronts these issues by helping to fine-tune development, placing a spotlight on excess costs and inefficiencies and calling for even minor improvements in process and tooling that, at scale, can have a substantial impact on the organization's performance. Many successful software companies—recently Dropbox and Netflix, among them—have built dedicated DPE teams to better manage product cycle times and improve developer efficiency.

Manage customer change and be prepared to say no. It is imperative for software companies to avoid falling into a customer trap when building new product lines and features, and when upgrading their products to run on modern and more accessible platforms. Software companies have numerous large customers that are either looking for one-off bespoke features or are not prepared to change their technology stacks or to adopt new platforms across their organizations. In both cases, many software companies divert their resources to keep one or a few customers happy and fail to fully understand the

opportunity cost and the longer-term impact of doing so on R&D efficiency and competitive advantage.

> Not all revenue is good revenue. Paradoxically, it pays to not listen to customers sometimes.

To avoid this trap, software companies must do two things. First, they need a clear process to triage incoming customer requests and they must be prepared to say no. This means avoiding altering product development roadmaps with customized changes requested by individual customers—no matter how valuable those customers are. Instead, from a revenue growth perspective, they need to look beyond immediate customer demands and determine if a customer's request will help expand the TAM (total addressable market) or help drive better economics for other customers. On the cost side, they need to consider the total expense of building custom features, including the recurring cost to maintain and support these features. Most importantly, they need to understand the total opportunity cost and business risk of diverting resources away from other items on the roadmap. Not all revenue is good revenue, and, hence, paradoxically it pays to not listen to customers sometimes.

Second, when modernizing or releasing new architectures, software companies should clearly identify the revenue at risk for different migration and modernization scenarios and be open to leaving customers behind. This assessment should consider, among other things, potential loss of certain customers and possible shifts in market influence as well as impact on pipeline and new opportunities. Armed with this knowledge, companies should create an incentive model to encourage uptake of the new software models. These can be carrots, such as discounts on the new products or assistance from software company experts to help a top customer migrate, step by step. In other cases, these may be sticks, penalizing customers who refuse to shift to the new product with higher prices for legacy versions. That may sound overly punitive, but companies cannot afford to scale

development only half-way; it is too costly to maintain hybrid versions of a product, and the negative impact on RDI demonstrates that it will reduce future growth. Sometimes, leaving some customers behind as a company embraces change is the only option.

Many leading software companies—Adobe, Ariba, and Salesforce, to name a few—have successfully transitioned their entire customer bases into SaaS-based products (or in Salesforce's case, transitioned partners and customers to a new architecture) following the approach we outlined here. All three took an aggressive stand, frequently communicating to customers that they want their business but only if they are willing to move to the new platforms. In essence, their message was: "We want to solve your problems, provide the features that you want, and improve your experience with our products. But we can only do that with a more modern product emerging from a more efficient R&D initiative." Importantly, these companies planned carefully for the changeover; for example, Salesforce ran the old and new versions of its software during a three-year migration timeline.

Software companies face a host of challenges even in their most routine development efforts, and competition is more intense now than it has ever been. Spending on R&D has always been seen as a way to overcome some of these obstacles and market leadership was dependent on these expenditures. Yet, while R&D is still critical to success and differentiation, knowing that R&D doesn't scale with company size will certainly come as a disappointment to many companies.

Fortunately, there are relatively straightforward answers to address the difficulty of scaling R&D: adopt a long-term product investment strategy while maintaining an effective portfolio in the short term; track and repay technical debt to drive continual architecture modernization; break organization silos, by investing in shared services and product management; invest in product scale multipliers; and manage customer change, being prepared to say no. These steps will not only generate again significant returns from R&D but make software companies feel more confident about coming out on the winning side of innovate or die.

Authors

Pranay Ahlawat
Partner & Associate Director
Washington, DC

Clark O'Niell
Managing Director & Partner
San Francisco - Bay Area

Archith Mohan
Project Leader
San Francisco - Bay Area

Ted Wiles
Associate Director
New York

1 We analyzed 191 software companies—approximately 70% from the United States, about 10% from China, and the rest from 13 other countries. Of the 191 companies analyzed, 187 (~98%) are public.

ABOUT BOSTON CONSULTING GROUP

Boston Consulting Group partners with leaders in business and society to tackle their most important challenges and capture their greatest opportunities. BCG was the pioneer in business strategy when it was founded in 1963. Today, we work closely with clients to embrace a transformational approach aimed at benefiting all stakeholders—empowering organizations to grow, build sustainable competitive advantage, and drive positive societal impact.

Our diverse, global teams bring deep industry and functional expertise and a range of perspectives that question the status quo and spark change. BCG delivers solutions through leading-edge management consulting, technology and design, and corporate and digital ventures. We work in a uniquely collaborative model across the firm and throughout all levels

of the client organization, fueled by the goal of helping our clients thrive and enabling them to make the world a better place.