McKinsey&Company

Chandra Gnanasambandam,
Martin Harrysson,
Rahul Mangla, and
Shivam Srivastava

An executive's guide to software development

High Tech February 2017

This essential capability is a blind spot for many nontech leaders.

In his 2013 message to GE shareholders, CEO Jeffrey R. Immelt wrote, "We believe that every industrial company will become a software company." Last year, he doubled down, moving GE's corporate headquarters from Fairfield, Connecticut, to Boston, in large part to lure world-class software engineers in the area.

GE is not alone in upping its bet on software-driven innovation. Today, a Tesla car has more lines of code than macOS or the Windows Vista operating system. However, the fact is that many companies that have made their fortunes outside of high tech—in medical devices, retail, and other analog industries—have been slow to catch on to this game-changing shift in what drives sustainable innovation, the shift from creating physical goods and experiences to smart software development.

Despite the mission-critical nature of software, it gets surprisingly little attention in the C-suite. Even those who have built decent software-development capabilities often have done so on the cheap; software executives are rarely given a seat at the table of top management, and software strategy is often determined three to five layers down the hierarchy. Companies pay a steep price for dismissing software's importance. These include the following:

- Vulnerability to tech-based disruption. Increasingly, business models are being disrupted through tech-driven innovation—just ask Uber's competitors in the taxi business.
- Subpar user experience and churn. With the pervasive availability and use of high-quality applications on mobile platforms, the customer's expectations have been reset. It's hard to imagine a successful business without a strong online and mobile presence.
- Higher costs and lower margins. Beyond customer experience and differentiation, software is pivotal in helping optimize operations and rein in costs. Global freight companies like FedEx extensively use technology to optimize supply-chain operations.

Fortunately, leaders in all industries can learn from the last two decades of software innovation and adopt the processes, tools, and organizational structures that have proved to be most effective.

To make software an advantage, executives need to be fluent in leading software-development practices and carefully determine how software is integrated into the organization. Most important for executives to get right from the start, however, is making software development a strategic priority, not an afterthought.

Understand leading software-development capabilities

The innovations behind software are just as critical as the software itself. Executives don't need to code (their developers may insist they don't!), but must understand leading development practices to determine the right approach for the company. According to McKinsey's software-maturity diagnostic framework, 15 practices across five stages define the software-development life cycle, and world-class companies typically excel in a majority of these areas (exhibit).

Setup decisions

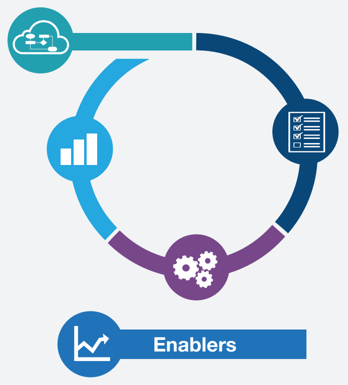
The strategic choices about how software is set up—where it is hosted and the underlying architecture—are one-time decisions with long-term implications. Such decisions should

Exhibit

Fifteen practices help define a world-class software-development organization.

Setup decisions that guide the strategic road map
1 Cloud-migration path
2 Platform choice
3 Microservices/container architecture

Product-management practices to aid in product conceptualization and design
4 Product-management excellence
5 Human-centric design



Product-delivery practices to ensure quality delivery
10 Analytics and use of telemetry
11 A/B testing
12 Community-driven development

Product-development practices to build and test quality solutions
6 DevOps (CI/CD[1])
7 Test automation and TDD[2]
8 API[3]-based architecture
9 Productivity and quality

Enabling elements to plan and operate
13 Portfolio management and product economics
14 Talent and governance
15 Product security and risk management

[1]Continuous integration/continuous deployment.
[2]Test-driven development.
[3]Application programming interface.

McKinsey&Company

be made with the same degree of rigor as any capital investment that is expensive and difficult to unwind.

- Cloud-migration path. With cloud technologies more pervasive than ever, organizations not already in the cloud need to chart a path to get there. One course is fast and direct: companies can create an extensible architecture that uses the on-premise code base as the core, with cloud-based applications developed from scratch. This approach is ideal to achieve a shorter time to market. Alternatively, a clean-sheet approach is better if the company plans to make the cloud the primary medium of delivery and the customer use case for cloud solutions differs significantly from the on-premise offering.

- Platform choice. Coupled with the selection of a cloud migration path is the adoption of a cloud platform. There are a variety of choices available that range from virtual private cloud environments to platform-as-a-service offered by established cloud players such as Amazon Web Services and Microsoft Azure. Four key factors drive the platform choice: commercial terms, such as the up-front cost of onboarding, recurring expenses, and contract flexibility; ease of use, including the ability to onboard quickly and support ongoing management; platform features, such as data complexity, compliance flexibility, and platform architecture; and data-sovereignty considerations based on where data reside.

- Microservices/container architecture. It is imperative to maintain a clear architectural road map across the portfolio of products. Successful organizations use data to select containers among open-source, cloud platform, and internal options (for example, J2EE app server).

Product-management practices

Effective product management is vital in the software-development life cycle. The function (and demands from product managers) have dramatically evolved in the cloud and mobile era.

- Product-management excellence. Expect product managers to act as the CEO of the products they manage, meaning that they own both the six-month feature road map and the three-year strategic plan. To meet this task, most leading software companies now require that product managers have both deep technical expertise and business acumen. For the cloud era, world-class product managers tend to be deeply technical people who are business savvy; this is in contrast to the on-premise world, when product managers were tech-savvy businesspeople.

- Human-centric design. Design thinking has taken center stage in building products that users love and admire. First, product design is shaped by insights from user research, customer-journey analysis, and storyboarding. The design concepts are then constantly iterated upon with customers. Finally, design execution is made central to the development process, with user-experience designers included as a core part of the development team.

Product-development practices

As software delivery has moved from multiyear releases to daily updates, software-development practices have evolved to focus on building high-quality software at an increasingly fast pace.

- DevOps. DevOps is the next frontier in the evolution toward increasingly agile development methodologies. In a DevOps model, engineers have extensive operational responsibilities to enable the release of production code. Companies need to master five core-competence areas to achieve DevOps at scale. These are continuous integration and delivery, automated testing, self-service access to infrastructure, automated performance management, and infrastructure that can scale automatically.

- Test automation and test-driven development. By automating testing and integrating it into the development process, teams create high-quality code that meets business requirements and can be deployed quickly. In test-driven development, test cases that describe user requirements are written first and are then applied immediately to test new code. For high-value test cases, particularly all regressions tests, tests are automated to ensure the quality of code in important areas.

- Architecture based on application programming interfaces (APIs). Historically, companies have suffered from building and maintaining "spaghetti code," which is as messy and difficult to manage as overcooked angel-hair pasta. An effective API-based architecture solves this problem and instead provides an extensible framework of building blocks that can be used to compose powerful applications. Like Legos, such blocks are easy to separate, update, and then replace. Effective API management extends beyond the initial design and covers life-cycle management and tracking, which is addressed by emerging providers such as Apigee and MuleSoft.

Productivity and quality

Once thought impossible, measuring software-development productivity is becoming mainstream, with "complexity points" being one of the emerging standards for evaluating software productivity and quality. Robust measurement also enables better forecasting of the effort required for new projects. An analysis of more than 1,600 software teams shows that top-performing teams significantly outperform in all aspects of software development. Top teams beat others to market with fewer people and defects.

Product-delivery practices

Cloud-based development has also enabled a more mature set of product-delivery practices that allow companies to gather more data than ever before, engage users on live experiments, and leverage the open-source community for faster development.

- Analytics and telemetry. Cloud-based delivery generates real-time data with deep granularity across the product portfolio, enabling a variety of uses.

For example, a rich data set allows product managers to make fact-based decisions on features and capabilities. Performance data can be analyzed to estimate usage trends and predict periods of high activity and stress on the system. Additionally, data-driven insights can help identify sales opportunities that can be delivered to the customer in real time. Finally, anonymized data from multiple users can also be used to create industry benchmarks that would be invaluable to customers.

- A/B testing. The capability to test different variants of functionality in real time with end users is now mainstream with cloud-enabled software. Leading companies integrate A/B testing practices into the software-development life cycle to ensure that development teams get feedback from users early in the development process.
- Community-driven development. Tesla has created a software platform with more lines of code than Windows, but with a fraction of Microsoft's software-development capabilities. How did Tesla achieve this without an army of engineers and three decades of experience? Through the extensive use of available and mature open-source software. Companies are now assembling capabilities through available libraries rather than writing code from scratch.

Enablers

To tie it all together, organizations require a set of enabling functions and practices that, while nontechnical in nature, are imperative to building effective software.

- Portfolio management and product economics. As the volume of software assets and capabilities explodes, there's an ever-increasing need for better portfolio management. There are five classic elements of portfolio management that should now be applied to software: market attractiveness, strategic positioning, investment analysis, risk assessment, and investment allocation.
- Talent and governance. Organizational structure and governance have evolved alongside technology in recent years. Most software-development teams are now structured in "pods" that bring together user experience, product management, DevOps, quality, analytics, and security resources. At the same time, governance now distributes decision rights across the pod, with the product manager acting as the CEO of the product.
- Product security and risk management. To build a secure product, security and risk-management thinking must be incorporated across the product-development life cycle. This implies that security transcends secure-coding practices. It includes involving a security champion in the DevOps team from inception, building a secure customer experience, and investing in tools and hackathons to identify security issues early in the development cycle.

Summary

The trends are clear. Over the last 20 years, the number of top-100 product and service companies that are software dependent has doubled to nearly 40 percent. Revenues from digitized products and channels are expected to exceed 40 percent in industries such as insurance, retailing, and logistics.

But the message here is not just that software matters or that it is increasingly found in things where software never existed before. All executives worth their salt already understand this in their daily lives, every time they drive their computer-controlled, high-performance sports sedans or take a fitness-tracking wearable on a run.

The point is that the C-suite has to take a more active role in how software is developed and make investments to build world-class software-development practices in their organizations. A modern company with any intentions toward industry or category leadership must be a great software company at its heart. Leaders of these firms need to have a secure understanding of how software development works and how to create an enabling organization around it.

It's not too late to get up to speed on software, but time is running short. The arrival of cloud technologies and the fast-cresting Internet of Things wave are two unstoppable forces promoting digital capabilities. Competitors that have already made the digital transformation are busy at work building tough-to-overcome competitive advantages. The next move is up to you. □

Chandra Gnanasambandam is a senior partner in McKinsey's Silicon Valley office, where **Martin Harrysson** is a partner, **Rahul Mangla** is an associate partner, and **Shivam Srivastava** is a consultant.