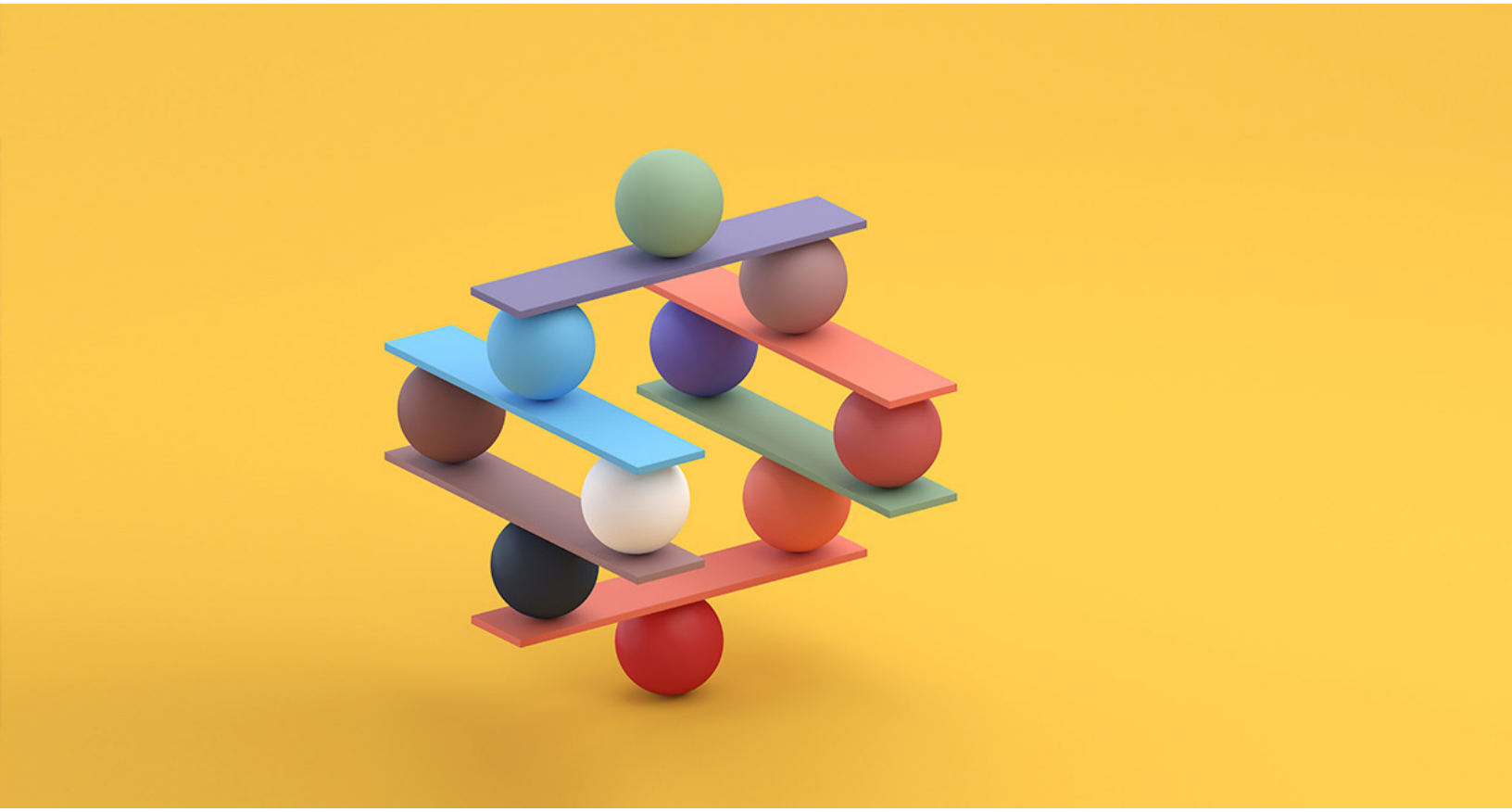
McKinsey Digital

The ERP platform play: Cheaper, faster, better

Enterprise resource planning upgrades can be expensive and complex—and unavoidable. A product and platform approach can manage costs and improve outcomes.

by Oliver Bossert, Florian Nocker, Christoph Schrey, and Murat Soganci



February 2023

Upgrading an enterprise resource planning (ERP) system is one of the biggest and most expensive decisions IT leaders will make. It can often cost as much as $500 million, last several years, and determine important elements of the business operating model for the next decade.

Before jumping into that decision, CIOs could benefit from understanding how tech companies approach changes to their system landscape. These tech companies value speed, flexibility, and scale to create value for the business. That means they make technology decisions that maximize freedom and independence for their developers by reducing system dependencies and complexities. As we laid out in our article "The platform play," digital-native companies deliver that independence by organizing their tech around modular products and platforms that run as a service.[1] This approach enables teams to make the best decisions for the product or platform they manage.

Contrast that approach with what happens at many incumbents, which labor under enterprise-wide systems where complex dependencies make independent decision making virtually impossible.

Consider the case of a large pharma company that used a single ERP system to provide distribution and warehousing despite the fact that distribution was a strong competitive advantage and needed to be flexible and responsive, while warehouse management was a basic commodity. The tight coupling between these two business domains in the ERP system meant that the changes needed in distribution were limited by system dependencies with warehouse management. This reality is a significant contributor to the accumulated technical debt.

It doesn't need to be that way. By focusing ERP upgrade efforts on the modules *within* the system rather than on the entire system and by understanding what matters for driving business value, CIOs can reduce dependencies, spend less, get more, cut back risk, and do it faster.

From ERP-centered thinking to a modern platform organization

The first lesson to learn from digital natives is that they set strategy first and design their platform architecture second. With the strategy clear (for example, increase the number of new customers and reduce customer churn), they follow a relentless logic in identifying "products," such as customer journeys (buy the product, find a store, get info on a product), that can drive the strategy.

With those in place, they then identify the platforms (such as user authentication and product comparison) needed to deliver those products. For each of those platforms, companies then set up a team that is in charge of the platform's outcome and performance and will ultimately also decide if it uses functionality that already exists in the ERP system.

This product and platform approach underscores two clear principles: first, treat the ERP system as a sum of capabilities rather than a monolithic stack, and second, the product drives the decision about what parts of the ERP system to use, not the other way around.

Taking the step toward a product and platform operating model is significant and requires a mindset shift for most IT organizations. Traditionally, companies have been focused on buying ERP solutions and managing the vendor and the system integrator to do the customizations. While this is still good enough for areas where one relies mostly on standard processes, it is not sufficient for areas where companies require something tailored to their needs. Most ERP vendors understand this and have themselves started to push for more modularity and a lean core. In the meantime, legacy IT has typically addressed the issue by building on top of the ERP system, leading to significant complexity in managing any changes.

The implication of this shift is that IT will need to be more hands-on in managing their ERP systems. This means developing deep engineering

[1] Oliver Bossert and Driek Desmet, "The platform play: How to operate like a tech company," McKinsey, February 28, 2019.

skills, actively managing system complexities and dependencies, and working closely with the business to ensure changes generate business value.

Determine what parts of the ERP system add value

With clarity around the strategy and a focus on a product and platform operating model, the next crucial step is to determine which elements of the ERP system directly support the business's strategy. At a high level, this value analysis divides the functions and capabilities within the ERP system into two buckets:

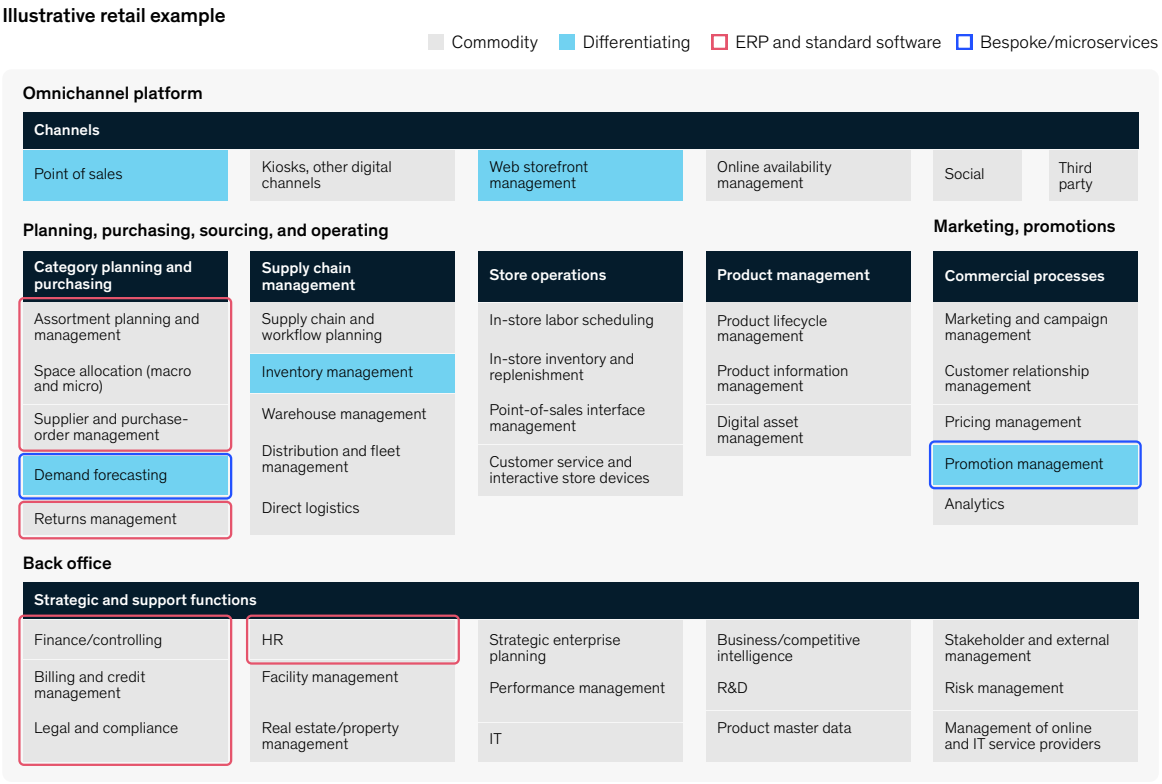
— In one bucket are the differentiating elements that deliver value to the business. For a retailer that wants to provide the fastest delivery, for example, that would mean prioritizing fulfillment and logistics capabilities. In many cases, those capabilities are delivered through microservices and are fully independent of the ERP system.

— In the other bucket are commodity functions that are not core to driving competitive advantage. In many cases, these functions include legal or property management. The ERP advantages in providing stability, tracking, and capabilities-management functionality are sufficient. If industry standards are applied, the company is often benefiting from innovations of the vendor and creates value without deviation from the standard. Any customizations created by IT need to contribute enough value to offset the work needed to maintain them.

The result of this sorting exercise is a capability map of differentiating and non-differentiating ERP capabilities and how they are organized (Exhibit 1). While most of the classifications can be made at the highest level, there are some areas that have to be split up even further. As shown in Exhibit 1, for example, the majority of assortment management is considered a commodity function, but the demand forecasting is where this company wants to differentiate itself from its competition.

Exhibit 1

Map business capabilities and identify value.



A well-designed capability map will also help to determine which cluster of modules needs to be upgraded based on value to the business.

This view provides a powerful counterbalance to the prevailing norm where the vendors, rather than the business, define the boundaries of functionality and modernization needs. This situation is often reflected in the fact that, in many incumbent companies, even non-IT stakeholders talk about the "ERP vendor X" upgrade project rather than the "finance" or the "supply chain" modernization project.

Focus on lowering cost and risk for upgrades to the low-value parts of the system

For the first bucket, differentiating ERP elements, CIOs should push for immediate upgrades. But for the second bucket, non-differentiating ERP elements, they should sequence upgrades thoughtfully to manage for cost and risk. In this way, upgrading is transformed from a single multiyear project into a series of smaller software projects that each last a couple of months. This scope reduction lowers the risk (small project = small risk) and puts off spend that can be better allocated to transformation programs that drive value quickly.

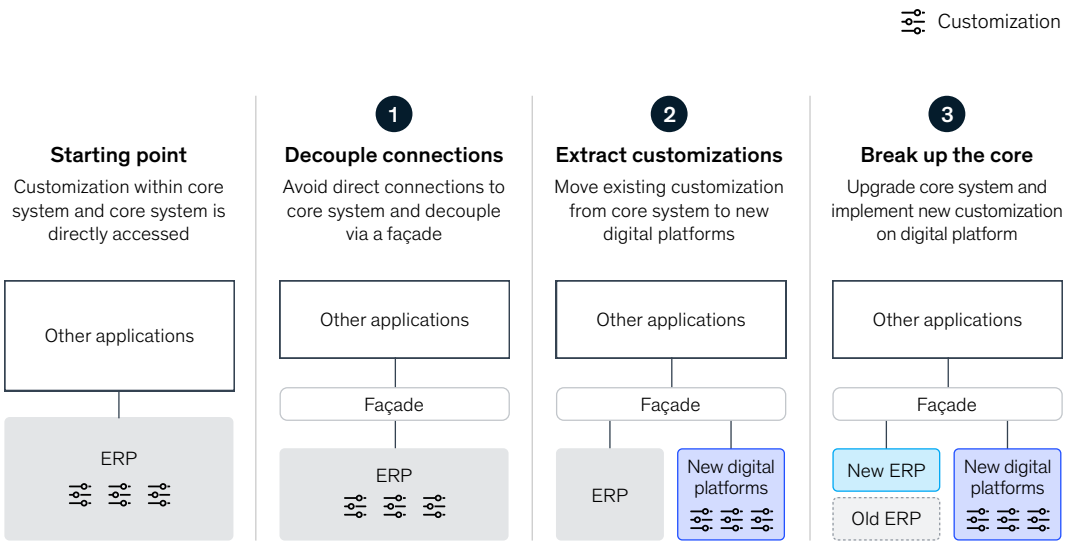
We have found that IT can capture these advantages by taking three steps to reduce the complexity of the monolithic ERP setup (Exhibit 2).

1. Decouple unnecessary connections

Replacing core systems can be a daunting task because of the variety of connections to other applications. Some of these connections help perform standard functions (for example, accounts payable), follow all the architecture guidelines from the vendor, and are easy to maintain. These connections are doing the jobs they were designed for and should not be touched. However, there are often many connections between parts of the system that are either workarounds or ad hoc solutions that are in place for a wide variety of reasons, such as a developer's decision that it was easier to directly access a database by creating something bespoke rather than using a modular interface. The sheer volume and uniqueness of these connections make any modernization effort complex and time consuming.

Exhibit 2

Take a technical approach to clean the ERP core.



To reduce this complexity, the first step is to create a new layer between the core system and the applications it connects to. This is often called a façade. All new connections will go to this façade layer via APIs that access data from the ERP system. In this way, the myriad connections are decoupled from the core system. This provides the big advantage of being able to make changes within the system, such as implementing modular architecture facets, without affecting all the connecting applications. The façade can be developed in less than a year. It doesn't need to be 100 percent perfect; it just needs to be functional.

But developers won't use even a good façade unless there is effective governance to enforce its usage. One way to do that is to make it easy by, for example, allowing product teams to access core features without going through lengthy approval mechanisms and waiting for someone in the core team to build an individual interface. In addition to such "carrots," "sticks"—such as penalties for those who do not follow the new protocols—may also be necessary.

2. Extract the customization

Most core systems have been customized to death, from complex reporting functions to bespoke access protocols. Each customization needs to be migrated—and often remediated in some way—to the new environment, which can be risky because of the corresponding complexity. To address this issue, companies need to build a digital platform (generally in the cloud) that can be accessed through microservices. The number and function of the digital platforms can vary; some companies, for example, will create one platform for customer-facing functions, one for supply chain, and one for the ERP system itself. The platform becomes the place to which customized functions can be moved and where new code is developed.

It's important to assess which customizations are most important. This process inevitably uncovers many customizations that are no longer needed and can be removed, thus simplifying the upgrade.

3. Shrink the core

Once customizations have been removed from the core, it's necessary to start to shrink the core itself to its most necessary functionality.

This is essentially a disaggregation process that removes the many intricate connections that are often built into large systems. In this way, developers and engineers can then replace or improve a specific functionality without affecting other parts of the system. This process generally includes cleaning up the code base as well, making it easier to understand and, therefore, to fix or replace.

As noted earlier, this disaggregation process does not need to touch those functional domains, such as accounting or controlling, that perform standard operations within the ERP system. Only those functions that are often part of a tightly coupled core for no functional reason, such as warehouse management, demand forecasting, or transportation, are candidates to reside outside of the core ERP system.

ERP upgrades are massive, complex, and necessary, especially as business pressures increase and cloud and ERP providers roll out new software and services. By taking a product and platform approach, however, companies can prioritize upgrades that create value and de-risk the upgrades that don't. In this way, companies can better manage costs and improve outcomes.

Oliver Bossert is a partner in McKinsey's Frankfurt office, **Florian Nocker** is a partner in the Munich office, **Christoph Schrey** is a partner in the Chicago office, and **Murat Soganci** is an associate partner in the Tokyo office.