HOW-TO GUIDE

Configuration vs customization—how to tell the difference, and which one to embrace

Most businesses that are in the market for new software want a solution that does exactly what they need it to do. However, the more complex those requirements, the less likely it is that a piece of commercial software will meet those requirements out of the box. In order to get the desired functionality, the software must either be configured, customized, or a combination of both.

Which is which?

A customization is a modification of a software feature that requires custom coding and possibly some form of implementation. A configuration, on the other hand, uses native tools in the system to change its behavior or adapt to specific requirements without the use of code.

Customization is more invasive to the core application because it requires programmers to modify or write a program to do something that the software does not currently do. Customizations can be basic, such as writing a class file to meet a task, or extensive, such as changing the core application.

Configuration enables end users to clone screens, modify fields, and make other changes on the fly, without IT intervention or code manipulation, to make the software personalized, role-based, and more quickly useful—enabling greater efficiency.

What happens if I choose to customize?

A critical side effect of the heavy use of customizations is the inertia that they create around software upgrades. With legacy software, this typically surfaced when applying vendor-provided patches for minor updates, and it became a major concern for larger updates, which required comprehensive checks and testing. This cycle has left many customers—especially ERP customers—with difficult choices about when to upgrade and when to remain on a potentially outdated version of an application. Over time, these choices have a compounding effect, leaving many organizations hopelessly behind current revisions due to the sheer extent of effort needed to update customizations.

Modern applications rely on configuration instead of customization, so there's no code that needs to be maintained through upgrades. And with software-as-a-service (SaaS) offerings, the continuous upgrade cycles allow organizations to leverage the latest vendor capabilities without compromising unique functionality.

Is “off-the-shelf” good or bad?

The term “off-the-shelf” can have mixed connotations, as it is often associated with old generic business solutions that offered only rudimentary functionality under the guise of “plug and play” simplicity. But modern software providers with true configurability in mind invest a lot of development time up front. Their goal is to build industry-specific capabilities and provide extensibility frameworks that enable an extraordinary number of configurations of the current product—as well as even greater value through connections to other products. Companies that don’t want to pay for long-term implementations should look for solutions that provide most of their requirements out of the box and offer the flexibility to configure the rest.

What does “extensibility” mean in this context?

As businesses expand their ERP considerations from core capabilities to extended Digital Operations Platforms (DOP), the ability to connect to additional applications, both in-house and external, is critical to business agility. As new relationships or business functions, such as enterprise asset management or workforce management, join the overall operational requirements, ERP must support the full range of integration modes—from API services for in-context needs, to messaging queues for batch transactions. ERP strategy needs to begin with the assumption that this connectivity is mission critical and not just an afterthought.

What role does SaaS play?

In order to keep global industry functionality current and tailor any one organization's instance to their business needs, modern software vendors should be designing configuration options that don’t require code changes. Instead the underlying application can be seamlessly upgraded while maintaining the configuration data. New capabilities should be delivered with toggles so that organizations can choose when they want to deploy additional tools to end users.

What’s the definition of “modern” software?

In the context of software, “modern” means that organizations can take advantage of built-in tools and workflows that are tailored to how work actually gets done. Trying to achieve this alignment with legacy software often involved extensive blueprinting and customization to match software behavior to business needs. The integration of other applications to work together in support of these processes also required heavy customization. While these modifications did help, they also created an IT headache with the additional overhead required to execute business changes. And they often became critical points for potential failure in daily execution.

Modern software should be designed with reasonable baselines of industry needs and the ability to easily be configured the rest of the way to match unique requirements. This approach frees up tight IT resources and creates greater business agility as requirements evolve.

Don’t hamper your operations teams by relying too heavily on IT intervention and vendor customization, and don’t bog down your IT team when they could be engaging in more strategic activities. Take advantage of modern software’s configurability options to move ahead quickly on executing your organization’s objectives and mission.

LEARN MORE →

Follow us:

641 Avenue of the Americas, New York, NY 10011

INF-2295533-en-US-0420-2