syncron™

The 10 Pitfalls of Building Your Own Inventory Management Software

Risk factors in homegrown software to consider for enterprises with global inventory distribution

A seemingly good idea

Building your own inventory management software in-house may sound like a good idea initially. Your company gets to design the software to do exactly what you want and to match your unique technology environment. It also retains total control over the features developed and subsequent intellectual property.

Unfortunately, it rarely works as first conceived. Building industrial-strength software that's critical to the business almost always takes longer and is costlier and riskier than anticipated. In some ways, it's like building a house on your own. It is possible to do it and while some have succeeded, for most, the likelihood of getting it wrong is high and the continuing costs to maintain it unsustainable.

We've seen homegrown inventory management solution fails happen again and again over the 20 years we've been in the inventory management software business.
A company will call us with a request-for-proposal (RFP) and after evaluating the pros and cons – between building their own system or buying it commercially – they'll decide to build.

Invariably, one to a few years later those same companies will call us again for help. The project is behind schedule, is increasingly losing ground to the competition, is way over budget and/or the team is asking for more, or while a functional inventory management tool, it doesn't quite do what the business needs to meet stated objectives and costs for the project.

For most companies, the risks of building homegrown inventory software are at too great of a cost to the main business of the enterprise. This eBook examines the risks and perils of building your own inventory management software for enterprises with global distribution.

Large software projects are expensive to build

Pitfall 1

While initial development costs of building enterprise software in house are difficult to estimate, we have seen these costs can easily run into the millions. The drivers for these costs include research and development (R&D), scope, integration, cybersecurity, quality assurance, data migration, timing, and ongoing maintenance and update – along with the size and experience of the development team.

Worse yet, those considerations don't account for the inevitable coding defects, bug fixes, overlooked requirements, developer attrition. It also doesn't account for a finished homegrown solution which simply does not meet the functional requirements, scale, and/or usability of what was intended or originally scoped and promised. The larger and more complex the software development project, the larger and more often the challenges gap arise.

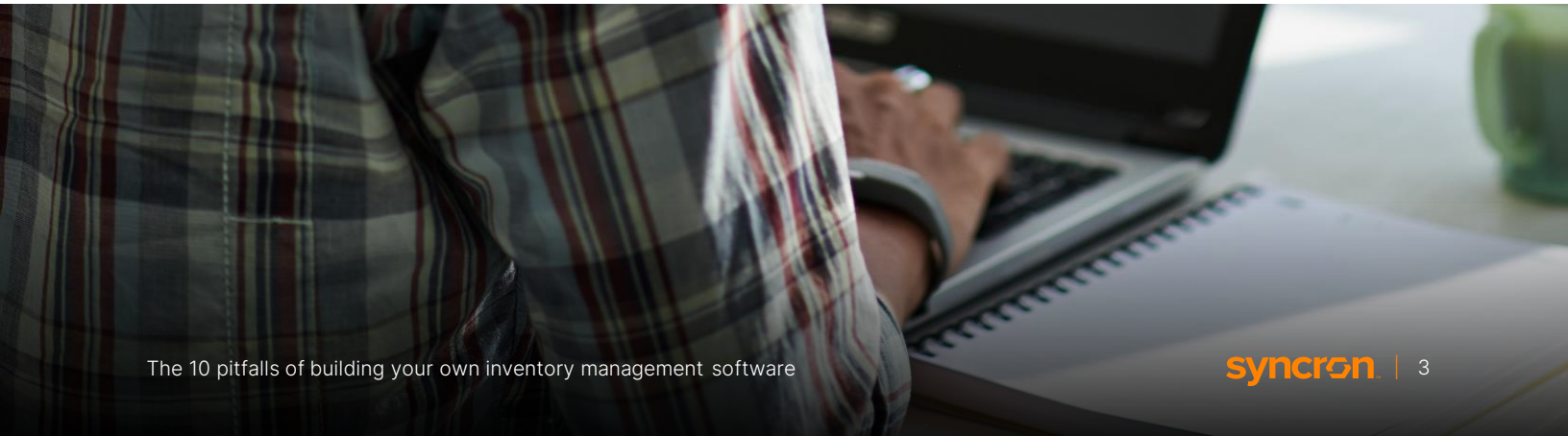
To provide some context, consider the following benchmarks:

- The average junior developer can produce an average of 100 lines of code per day or 26,000 lines of code per year. Assuming a conservative hourly rate for a U.S.-based developer of $150 per hour, the annual cost is $312,000 per developer. That's barely enough to cover a minimally viable product (MVP) – assuming the project goes flawlessly. Mature inventory management software involves hundreds of thousands of lines of code.

- A software CEO we spoke with, who has experience building custom enterprise resource planning (ERP) software, suggested looking at the engineering costs at software startups as a reasonable way to back into an estimate for a project developed in-house. He says small companies in this space "easily spend" between $2 million and $10 million annually on software engineering alone.

- A software pricing guide by Better Buys, a software review site, puts the cost of building a comparable ERP system for large businesses in a broad range of $1 million to $10 million.

Modern commercial off-the-shelf tools (COTs) hosted in the cloud eliminate many of the upfront costs in hiring and managing a development team in house. The COTs instead offer a pay-as-you-go subscription model that significantly lowers the initial price tag and makes the software costs predictable. It also changes the type of expense necessary to get started from a capital investment to an operating expense, which many businesses find favorable.



DIY software has higher operating costs

Pitfall 2

Total cost of ownership (TCO) describes the acquisition costs of software – whether procured or developed in house – plus the cost of operating it once it's in place. This is important because the initial cost of developing an application is just a fraction of the total cost.

Market research suggests 84% of the TCO attributed to DIY software is attributable to operating costs after the tool is in production. This includes development fixes, infrastructure maintenance, security patches, IT helpdesk, and training, among others.

What's more, IT operations costs are growing at a rate that exceeds initial technology acquisition costs. For example, in a report published in 2020, the consulting firm McKinsey & Company said the costs associated with software maintenance grew 40% from 2014 to 2017. No function in business can easily sustain such an escalation in costs.

Such figures are reminiscent of an older, but still relevant, anecdote McKinsey shared a few years earlier: a large retailer filed for bankruptcy after two major multi-year IT projects failed. These included a $1.4 billion IT modernization project and a $600 million supply chain management initiative.

Observers will point out that cloud environments have reduced the costs of physical hardware. That is true, however, most businesses can't match the buying power of a dedicated technology provider that purchases high availability cloud environments for hundreds of software deployments across its customer base.

For most companies, the operating costs of commercial inventory management technology, especially that which is procured under a software-as-a-service (SaaS) model, are far lower than attempting to maintain any kind of home developed software yourself.

Software takes time to develop

Pitfall 3

One factor that goes overlooked in building your own software is Time-to-Value (TTV). TTV is the amount of time it takes you to realize value from your product; it implies achieving the effectiveness of your investment. A robust software product to manage inventory is a complex undertaking and takes years to build, test, and deploy to a production environment. Often TTV can be quite long or illusive entirely.

Even when the software is complete and ready for prime time, the solution still must be implemented and accepted for use across a team. The key implementation tasks range from training and change management – to communicating with a distribution network and integrating the new software within partner IT environments or complementary platforms like ERP systems, to changing processes and the way in which users do their day-to-day jobs.

The time-to-value concept is important because it highlights subtle costs that aren't included in the acquisition or operating costs. For example, unused inventory sitting in the distribution network – often the catalyst for inventory management software – sits in the network while the software is being developed. That excess inventory can drive up costs by as much as 5-10% annually.

Commercial software is ready to be implemented on day one, and most software providers have a professional services team that has conducted hundreds if not thousands of implementations with deep levels of experience not only in deploying the systems but bringing users on board with training and enablement. These teams can have you up and running in a matter of weeks. Even complex deployments with a network of distribution partners and dealer management in a commercial software scenario can be accomplished in a relatively short amount of time.

The time-to-value concept is important because it highlights subtle costs that aren't included in the acquisition or operating costs. For example, unused inventory sitting in the distribution network—often the catalyst for inventory management software — sits in the network while the software is being developed.

5-10%

driven up costs due to excess inventory

Overlooked product requirements

Pitfall 4

While all enterprise software endeavors are challenging, software that optimizes parts inventory and availability is an especially tricky business, because gathering specifications is an intensive, tedious, and detailed process. You can spend months documenting requirements and there will still be use cases that are missed through the exercise. Some examples we've seen commonly missed in homegrown software include:

- Managing supersessions
- Algorithm-driven inventory simulation capabilities
- Forecasting for slower-moving demand patterns
- Distribution network integrations, and
- Single sign-on user authentication

Slower moving demand patterns is an example of an important edge case that is commonly overlooked in DIY inventory management software. Consider the low-volume-low-demand use case for a car manufacturer producing replacement engines. The manufacturer must decide how many of these replacement engines to build and put on the shelf at a dealer.

Let's say on average, they might sell one engine every three months. How many engines should they put on the shelf: zero, one, or two replacements? That's a hard decision to make and averages, by definition, can disguise important datapoints. Given the cost to make and carry an engine, it's also an expensive decision.

If they furnish the engine and don't sell it, they are carrying excess inventory, which ties up cash. If they don't furnish the engine, and a prospective customer needs one, that prospect will go elsewhere. The company stands to miss a sale and perhaps lose a customer for good.

When such requirements are discovered after a product has been built and deployed internally, it can be months or years before the resources needed to develop a fix or new feature are allocated. It's hard to go back to a leadership team with outstretched hands and ask for more budget – especially after an expensive project is supposedly completed.

Commercial software providers employ dozens of supply chain experts on staff who are exclusively focused on continuously collecting user requirements from hundreds of companies and testing what works and provides value. So, the chances are, a commercial off-the-shelf software product that's purpose built for inventory management will have better coverage of edge cases which are problems an in-house development team will often overlook.

Continued reliance on manual processes

Pitfall 5

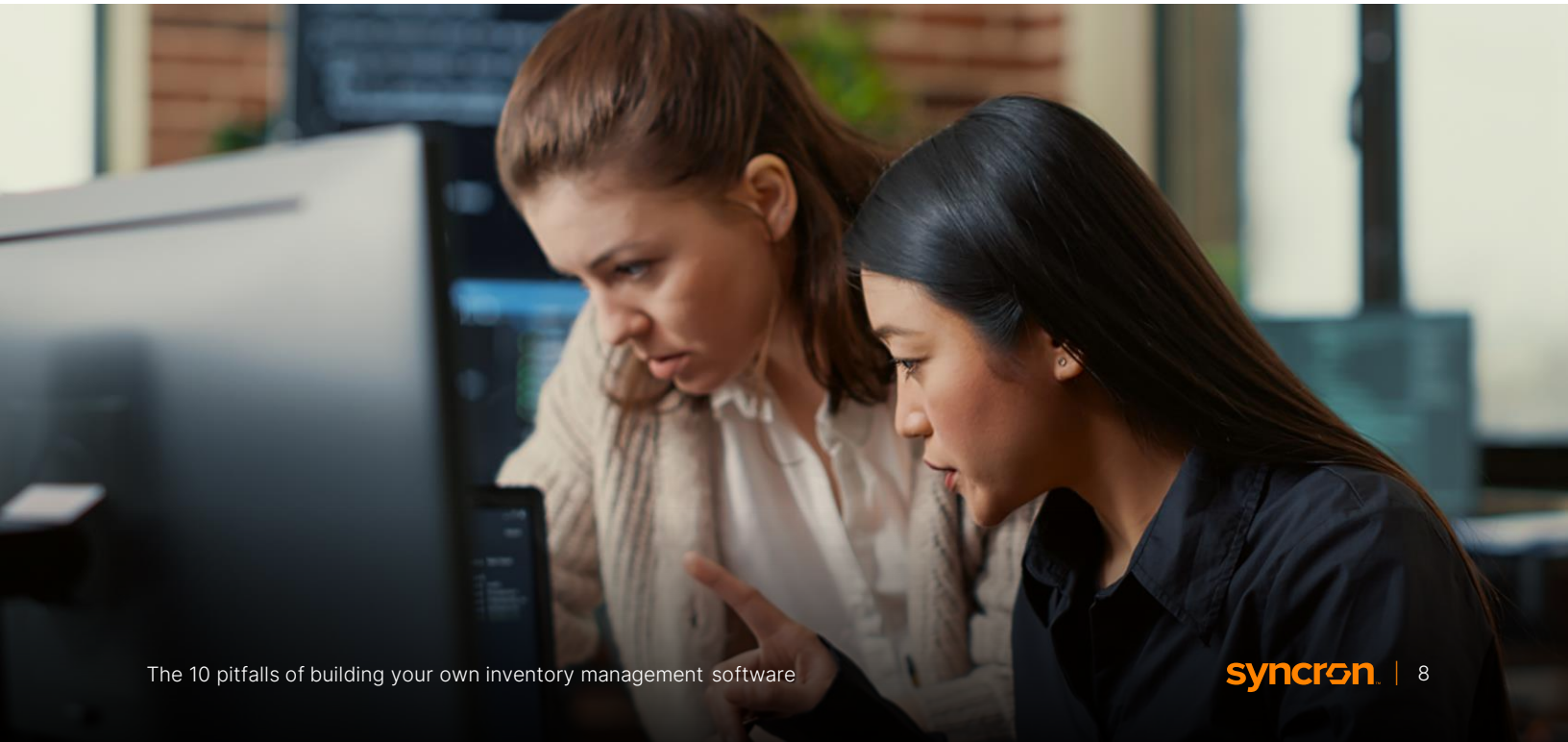
Homegrown systems often force a continued reliance on manual processes related to overlooked product requirements. As a result, and to get the job done, individual performers typically end up creating manual techniques to work around the gaps in software features and functions.

They might export data into a spreadsheet, run an analysis coded into a macro and then import the data back into the custom software. The process is inefficient, prone to error, and has version control issues. Moreover, the manual effort misses the productivity benefits of automation, which is one of the driving factors behind technology adoption.

Manual processes can also generate a single-point-of-failure in a crucial business process. For example, if there is only one person that understands how the macros work, and they go on vacation, or worse leave the company, there now exists a spreadsheet that no one else knows how to use. Manual processes also require special one-off testing and retrofitting when newly developed functions are deployed.

We see a lot of point failures when an enterprise attempts to customize its existing ERP software to provide ad hoc inventory management. However, these systems tend to be modularized and segmented, so the user ends up adding manual processes. For example, by processing forecasts in one component and exporting that to a spreadsheet, so it can only then be imported into another component for optimization.

Workarounds can't match tightly integrated and advanced features of industrial-strength commercial software that comes ready to go out-of-the-box inventory management COTs.



Falling behind on innovation

Pitfall 6

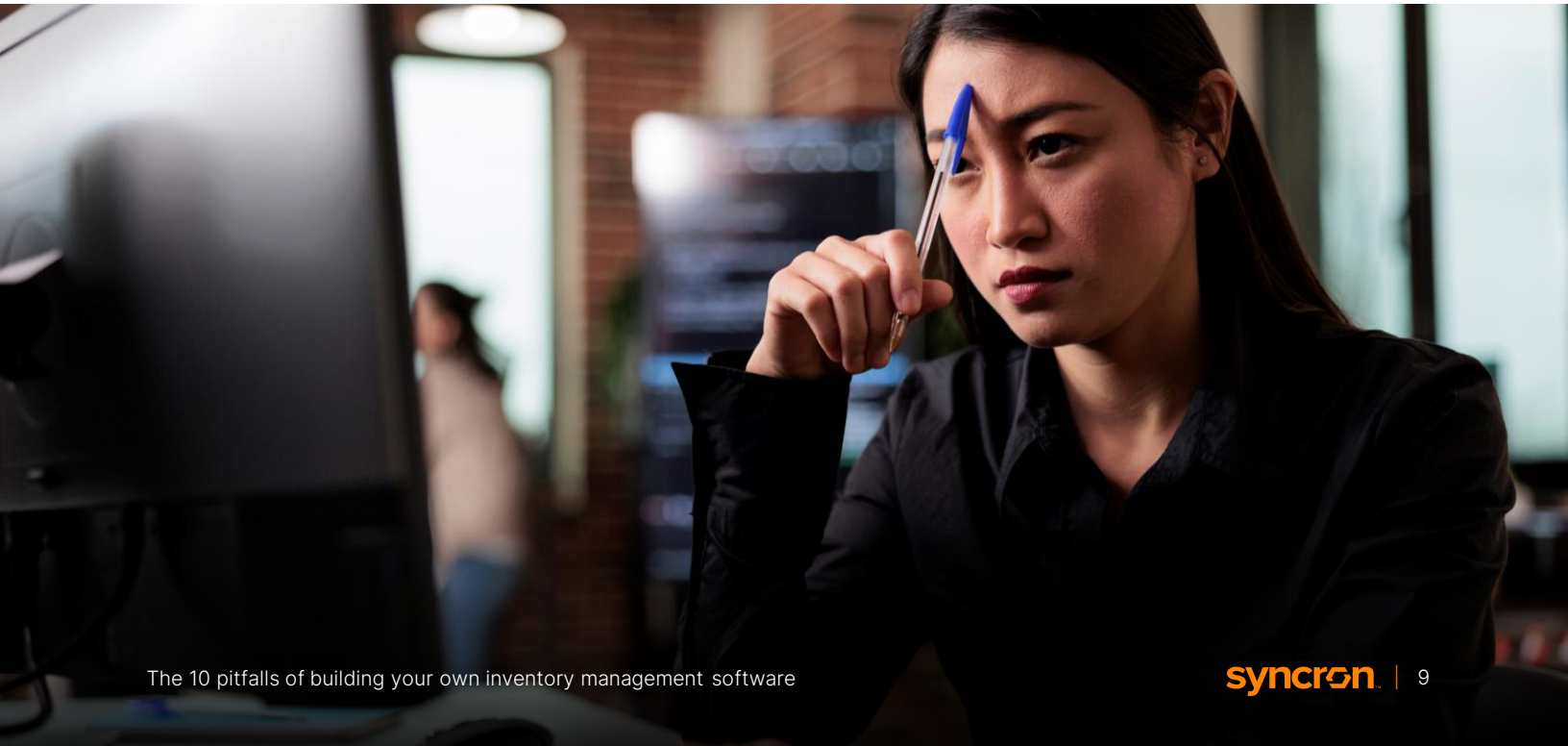
Keeping up with the accelerating pace of change in software and demand for innovation is a tall order. It's even harder if your core competency of your business function isn't software development and inventory management technology like forecasting. This is especially true for manufacturers whose core competency is producing physical products.

Businesses that build their own software regularly find their development resources are consumed by maintenance keeping the system patched together They don't have time or an R&D budget to continuously focus on new enhancements. These companies miss out on the benefits of introducing new technologies and expanded capabilities.

We've seen companies struggle with legacy software programming languages, too. For example, many have come to us with a product built in older and outdated languages. One company had a list of enhancements they wanted to make, and the budget to do it – but the IT department had tremendous difficulty in finding talent that could code in the C++ programming language.

One of the benefits of choosing a dedicated technology supplier for inventory management software is innovation. They have a dedicated research department and can evolve the product and keep up with current technology trends and meet evolving needs.

Another benefit of working with a commercial software provider is that they will typically have a wider community of current customers, who share common needs, requirements, and experiences. Peers from non-competitive vertical markets can meet in confidential forums or events, to exchange ideas and best practices – that you may not get when you go it alone.



Security and compliance

Pitfall 7

Software security has grown into a distinct technology discipline. It is a perennial game of cat-and-mouse with the defenders striving to keep one step ahead of bad actors. And the bad actors are increasingly sophisticated organizations, and in some cases, backed by nation-states.

Maintaining any software, including homegrown, requires a robust yet efficient IT change management process for fielding updates. For example, when a vulnerability is reported in an operating system, enterprises must test the patch before deploying it to a production environment to avoid unintended consequences, such as disrupting other systems. This sets off a race to patch known vulnerabilities before bad actors can exploit them.

The example we used was a reactive effort but note that the consulting firm McKinsey & Company also recommends preventive measures: “Enterprises need penetration tests and red-team exercises for their own homegrown capabilities to ensure that they are in line with security requirements; the same approach must be required of third parties as well.”

Enterprises must worry about their partners too. Disrupting the supply chain is a juicy target with many network partners, all with varying degrees of defensive layers. This provides the opportunity to gain a foothold in smaller, less well defended partners, and then move laterally across the network to higher-value targets, and perhaps encrypting the data and demanding ransom.

The cost of security incidents has skyrocketed in recent years. According to the 2021 Cost of a Data Breach Report, an annual assessment by IBM, **the average cost of a ransomware attack was $4.62 million.** Breaches took a total average of 341 days to contain.

Nothing today is completely foolproof but having the experience and technical chops to design software with security in mind is crucial. In addition, third-party providers often maintain dedicated resources to monitor emerging threats and develop countermeasures – like software patches – rapidly.

Inventory management software providers can illustrate domain expertise through national quality standards. One such example is the Service Organization Control (SOC) 2 Type II certification, which was created by the American Institute of CPAs (AICPA).

The standard is used to audit service providers, including software vendors, to make sure the provider is protecting your interests across five areas including security, availability, processing integrity, confidentiality, and privacy.

Taking on cybersecurity alone especially without it being a core competence is a steep risk. An outside software vendor who views security expertise as a key part their offering, with proven domain expertise, is a sound step toward acute risk mitigation in general.

The cost of security incidents has skyrocketed in recent years. According to the 2021 Cost of a Data Breach report,

$4.62M

was the average cost of a ransomware attack

341 days

in total on an average were taken to contain them

Human resources dimension

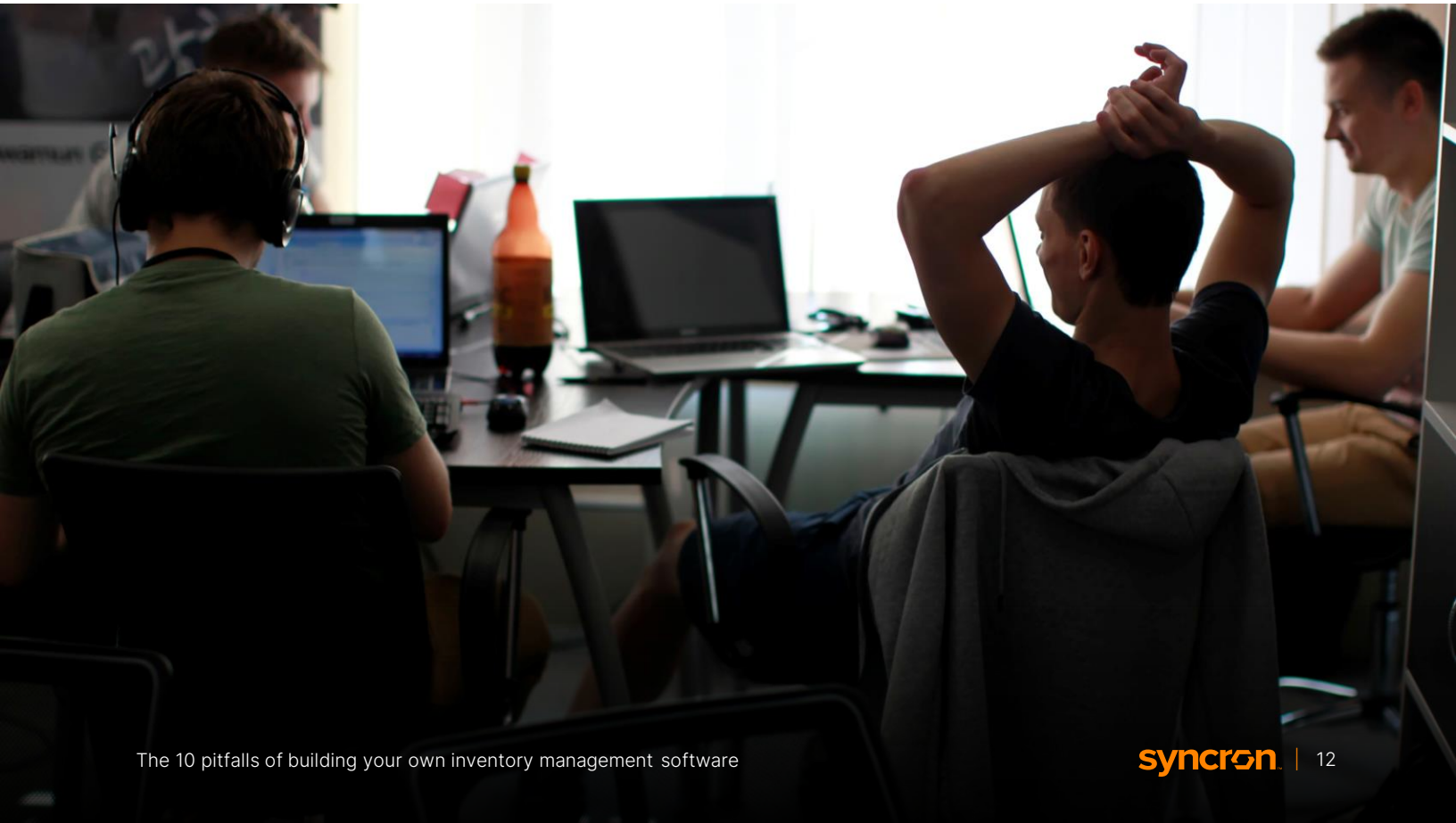
Pitfall 8

Technology can have a clear impact on human resources including recruiting and retention. Businesses that develop a reputation for investing in, and adopting, the best technology tend to attract like-minded talent. The reasons for this are simple: the technology skills they learn and acquire on the job are both good for their employer and individual career progression.

That's difficult to match with homegrown software. The technology skills employees acquire on custom software aren't marketable in the same way as skills learned using software based on supply chain industry standards. Employees realize that working on homegrown systems do not offer the same resume lift as working on systems recognized as leaders in their domain. Chances are, you'll have a tougher time recruiting and keeping talent.

A second risk in the human resources dimension is the flight risk of the functional architects behind the software you build in-house. These people develop institutional knowledge that is pivotal to the smooth operation of the process and technology. When they move on, it's difficult to capture and retain the lessons they've learned along the way.

Commercial software providers solve this problem because their focus allows them to attract the best and brightest technology talent from development to customer support. Their products also come with detailed user guides and technical documentation.



The sunk cost fallacy

Pitfall 9

People "commit the sunk cost fallacy when they continue a behavior or endeavor as a result of previously invested resources" such as time, money and effort. For example, "a person may have a $20 ticket to a concert and then drive for hours through a blizzard, just because she feels that she has to attend due to having made the initial investment."

We see this occur when it comes time to replace a legacy system – or more commonly when a business tries to customize an ERP system to provide inventory management features.

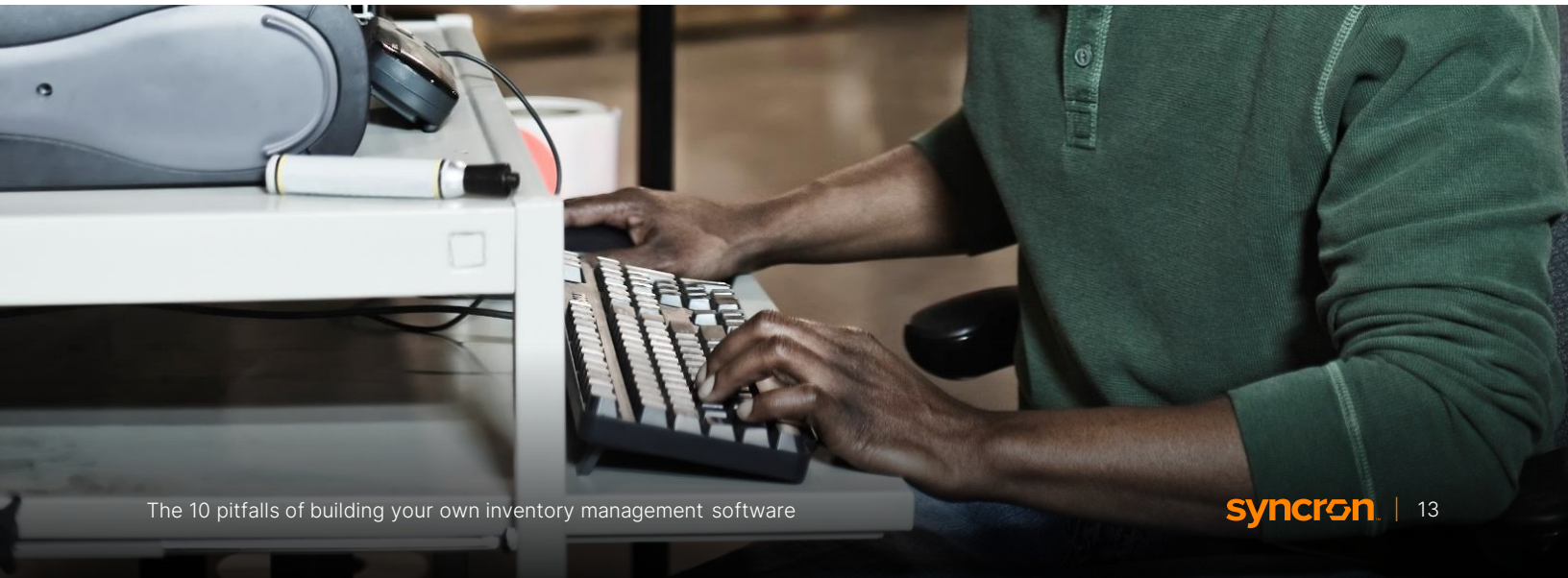
Why does this happen? Chief Information Officers (CIOs) have invested millions and perhaps committed their careers, to deploying a major ERP tool. When supply chain professionals approach IT for an inventory management solution, the CIO is prone to ask, "Why can't you do this in the ERP?"

They'll set out to customize the ERP to perform inventory management. When the features don't quite achieve what the end-user needs, IT continues to invest more and more resources to course correct. In our observation, they frequently fall short.

This becomes, at times, a bit of a political battle. The back and forth in development often leads to more work, not less for the inventory management professionals. The key to avoiding this scenario is to prove why inventory management can't be sufficiently performed in a customized ERP early in internal discussions.

In situations like this, prospective customers, usually a businessperson responsible for the aftermarket sales, such as parts distribution, will ask us how to navigate the situation. Our suggestion is a tried-and-true best practice in technology: request a proof of concept (POC).

A POC, or pilot program, is an alternative implementation approach that allows you to test functionality without making a large commitment. It will allow both sides to experience the challenges firsthand. It also allows a gracious retreat for senior IT leaders that avoids the perpetuation of the sunk cost fallacy.



Personal and departmental reputational risks

Pitfall 10

We've watched people and businesses make a comparison to commercial software and consciously decided to build inventory management software on their own. Inevitably, they run into unforeseen and unexpected complications.

Still, those leaders with a vested interest will insist success is just around the corner. The business continues to invest in the project, even as it limps along for a few years. Finally, frustrated by the lack of results, leadership steps in, cancels the project and initiates another look at commercial options.

Such outcomes harm the credibility of both the department and project champion. No department head wants that mark on their team's record, and no person wants to put a failed IT project on their resume.

A commercial provider diversifies such risks. The software is already built and has likely been implemented hundreds of times. The provider brings that experience with them which greatly reduces the risk of failure.

Buying versus building software

Has clear advantages

Homegrown software may be right in some situations but in most cases are not the right decision. They suffer from unknown and higher upfront costs, are generally more expensive to operate, maintain, and update, historically require a significant amount of time to pass before deployed (sometimes even years), require significant time to deliver a return on the investment made, and to not incur continuous and increasing ongoing costs. Even then, there's a high probability that the final product will be besieged with problems including bugs, missing requirements, and reputational damage from an underperforming technology tool.

"Building your own solution in-house...gives you complete control: after all, who knows your business, and its data, better?" wrote Deloitte, in a 2020 post.

As with anything bespoke, building your own solution inhouse comes at a steep cost. To do it successfully requires a number of resources only working on that project. A commercial software vendor has a huge efficiency advantage, in effect, a vendor spreads the development costs across a larger set of customers."

However, the Deloitte also noted in the same post, "Buying a pre-built tool should be cheaper, faster, lower risk, and will likely offer a complete, integrated, solution to a particular business problem."

While there are some unique companies with exceptional business needs – and the resources – to pull off a custom IT project of the size and scope of inventory management software, most will experience this to be impractical in the long run.

Purpose-built, off-the-shelf, cloud-based software, especially those commercialized under a subscription model, tends to be less expensive, faster to implement, and deliver greater benefits to the inventory management function and the overall business.

Want to see for yourself why the world's leading manufacturers choose Syncron for aftermarket service management solutions?

Simply complete this form and a member of our team will contact you within one business day. You can also email us at info@syncron.com.

Connect with us on social media: Twitter, Facebook, LinkedIn or YouTube.

For more information, visit www.syncron.com.
You can also email us at info@syncron.com.

Syncron empowers leading manufacturers and distributors to capitalize on the world's new service economy. With our industry-leading investments in AI and ML, Syncron offers the first, innovative, customer-endorsed, and complete end-to-end intelligent Service Lifecycle Management solution portfolio, encompassing service parts inventory, price, equipment uptime, warranty, and field service management. Delivered on the Syncron Connected Service Experience (CSX) Cloud, our solutions offer competitive differentiation through exceptional aftermarket service experiences, while simultaneously improving aftermarket business profitability. For more information, visit syncron.com.