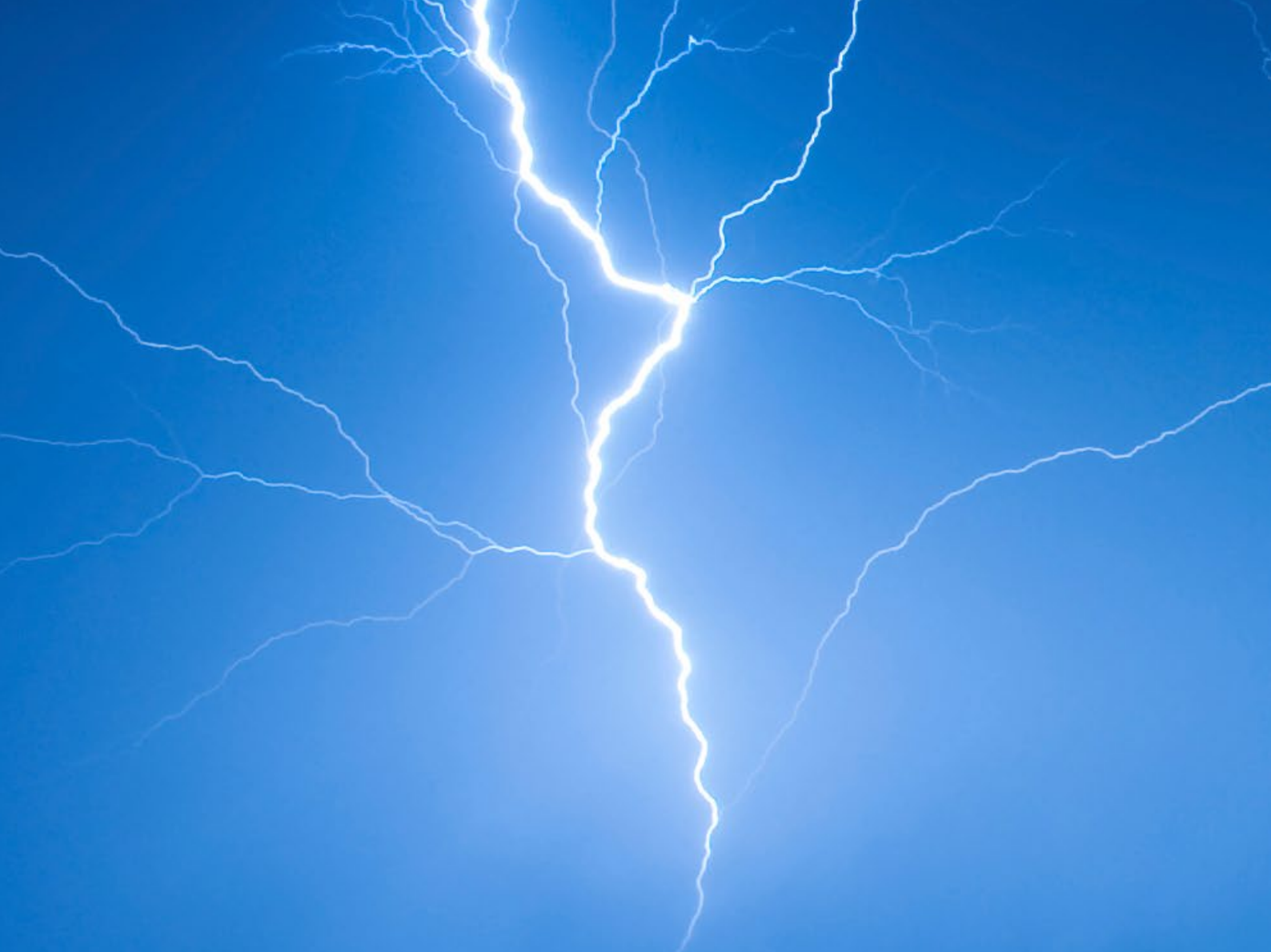


BUILD *vs* BUY

An insider's view of the considerations of building versus buying software

iasset com®

The Global Channel Ecosystem

As software continues to "eat our world"[1], there is a regular dilemma that crops up in organizations. When faced with inefficient and manual processes, the immediate reaction is: "So… how can we automate this?"

Introduction

Depending on the magnitude of the process, the next dilemma is whether a purpose built in-house developed application could solve the problem - or whether there is an outsourced solution that can address it faster, painlessly and more cost-effectively.

We think of the build versus buy approach to software as being similar to one's decision to build or buy a house.

There's no doubt there are some people out there who will always choose to build their own home. But the majority of people are likely to just buy one. Just like some companies will want to build their own software solution (whatever the downside risks) – many will want to buy a proven and effective commercial off-the-shelf (COTS) solution.

In this new world of software development (including open source frameworks and agile methodologies) there are many advocates who claim that building software is now much more accessible than previously. Accordingly, these promoters argue that the build option should always be seriously considered by companies looking to solve manual processing inefficiencies.

We whole heartedly agree.

But just like building your house, building your own software requires extensive planning, scoping, budgeting, project management and co-ordination, regulatory interface, issue resolution, on going software updates, maintenance and much, much more. Basically, the success of any build project really depends on these key factors:

1. Access to adequate skills
2. Available budget (time and dollars) to complete the project
3. Commitment
4. Connectivity to the great world

And like those two little pigs who built their houses of straw and sticks, building is simply not the recommended solution for everyone. Indeed it can be a disaster waiting to happen.

In this e-book, we argue that, just as there is always a time and a place for building, there is also a time and a place for buying.

At the end of the e-book we hope you will be in a position to make the wisest decision based on your circumstances.

1. Marc Andressen, Why Software is eating the world, 2011.

The Build Scenario

1. Design your own application

This is touted as the biggest advantage of the build option. Like selecting all your foundations, plumbing, interiors and furnishings for a house, it's great to be able to design a software with all the features, functionality, interfaces that address every aspect of your business process that is causing the issues and inefficiencies.

But like designing your house, designing the software to address process issues requires long term, flexible and scalable solutions. After all, you don't want to build something that doesn't work for your situation in two years time.

2. Control your own environment

People, and organisations, often like to be in control of their environment. When building your own software you can have complete control over the work environment, including planning, design, development (building), testing (installing) and support (maintenance).

This can be a double edged sword. You have no one outside of your team who can warrant the solution. When something goes wrong you often need to hire an expensive consultant. Under the build scenario, there must always be someone on the team who will be able to solve any crisis or issue.

3. Drive internal innovation

Testing one's own innovation abilities can be an exciting opportunity for many of us. In the context of software, Debbie Madden in an HBR article points out that "established enterprises need to innovate to keep pace with the more nimble, smaller start-ups".

She also mentions that "innovation by definition is inefficient", highlighting that the process is not always that easy and can result in lengthy time and resource consuming exercises.

Sound familiar? If you've ever spoken to anyone who's built their own house, it's likely that timelines have not been met, which we all know lead to significant and unforeseen cost over-runs.

Debbie Madden, 4 Tips for Launching Minimum Viable Products inside Big Companies, Sept, 30 2015.

As Otto von Bismarck once said: "Only a fool learns from his own mistakes. The wise man learns from the mistakes of others."

The Buy Scenario

1. Experience the lessons learnt by others

Provided you've done your due diligence, one of the incredible values of buying (either a house or software), is that someone else has already done the hard work and you are the beneficiary of his/her experience and output. Building in isolation means that you will be learning "on the job" and the outcome will only ever be as good as the skillset, knowledge and experience of your immediate team, which may or may not be adequate.

In the context of buying a software solution, you should be benefiting from millions of dollars of someone else's investment in code, and years of innovation for a comparatively low cost. The ongoing licensing cost provides you with continued incremental benefits from sharing the development resources with other organizations.

2. Access the industry experts

Let's face it, we can't all be good at everything.

In the same way that enterprise organizations have their core competencies, software companies do too. Software products wouldn't exist if they didn't solve a problem. By buying a solution, you have effectively tapped into the software company's knowledge, expertise and experience in solving the problem you are facing, which can rarely be supplanted by the internal staff of a company where this is not their proven skillset. In essence, you are fast tracking the learning curve and sharing the cost of the expertise.

The Buy Scenario

3. Scale your business (and software) quickly

Once you decide how big your house is going to be, it's not that easy to scale it up or down without significant cost, time or effort. Indeed, rather than building a new level, most people simply move to another house, which can bea waste of time and money.

It is the same for purpose built software.

An internally developed application faces similar challenges of future scalability. Purchased software solutions on the other hand are all about handling scaling and future growth. Remember, the software developer from whom you buy your application needs to provide scalable solutions to meet the requirements of their many customers. There is no point building a program that in a year or two cannot handle the increasing volumes of data that have been generated by the growth of your program.

4. Generate ROI faster

Any build project takes a long time to materialize any value. Whether it's building a house or software application, it will always come down to how much budget, time and effort you're willing to invest to achieve your desired outcome. The payback period for software development can be years – if at all.

Although there is an associated integration cost with any purchased software application, you will usually find that it will generate ROI much faster, as you will be the beneficiary of the software providers many efficiencies – proven through their experience.

We estimate that it can take up to 2-3 years of work for major enterprise process challenges to be adequately addressed via a software solution. By outsourcing the solution you should budget to see a return within weeks or months rather than years.

Other Cost Considerations

Staffing

It takes many skills to design and deploy a build project from scratch. In the software context, these skill sets are vast: back end, front-end development, data management, design, support, maintenance, quality control, copywriting for procedural documentation – the list goes on...

Recruiting or upskilling

It can be difficult to have every skill set required to complete a software build project. You either need to hire additional people or upskill existing ones. Regardless, this represents an investment in recruiting or training, which pushes up the cost of your project.

Training

Getting the right people trained is essential for the success of any project. In the software context, and depending on the level of complexity, it may take days of training for people to be fully independent and productive on the new tool. And even if you hire trained staff, you need to ensure they are kept up to date with a regular ongoing training commitment.

Employee turnover

What happens if one of your star tradespeople quits halfway through the project?

It's the same with IT solutions. If a critical staff member decides to move on half way through the project this can result in significant project delays and missed milestones, further adding to the cost of the project. It's like your builder quitting midway through your house building project. Good luck!

Technology

On-going support

Maintenance! Well, everyone has some form of maintenance. Build projects are typically very heavy on maintenance requirements, so it's important to consider this in the context of the software build.

Once an application is developed, companies often underestimate the costs required to support and maintain these types of systems. Systems need to be constantly re-engineered to support adds/moves and changes in business initiatives and often need to be completely reconfigured. If this results in downtime of a business-essential system, the loss of business can be catastrophic.

Other technology costs associated with maintaining a business-essential application include: system monitoring, planning and scaling, constantly evolving security requirements, documentation and more. With an outsourced solution, these are typically covered by your license fees.

So what if you've bought and there's a few things that just aren't quite right...

Customizing Commercial Software

Whether you have built your house from scratch or purchased it ready-made, you're likely to want to change at least something.

It's not that different with software. There are not many out-of-box solutions that can be designed as a one size fits all - because we all like different things. For organizations, they operate differently and have different processes that need to be accommodated.

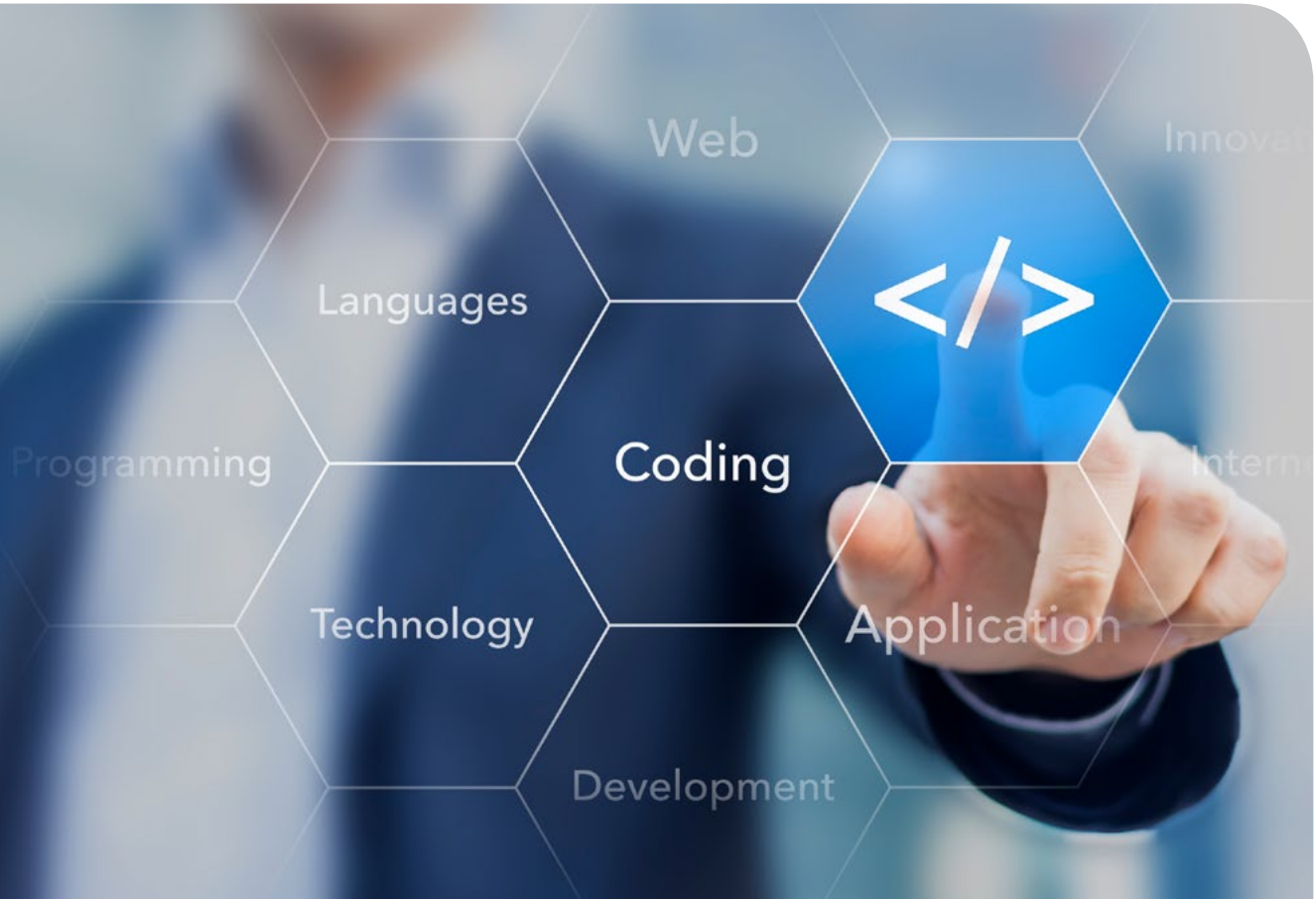
It's important to evaluate how essential the customization is to your particular process. Could you survive without additional custom work, or would it still be cheaper in the long-run to invest in some custom development, rather than build the whole thing internally?

On the flipside to customization, there are sometimes items we don't want or need when buying a software product.

As SaaS companies strive to make their product usable and meet the requirements for all of their customers, inevitably there are features that not everyone wants.

Ask what can be excluded from your deployment as some functionality can be easily "switched off". This may also help reduce cost or avoid confusion by having options that people do not use.

Keep in mind this functionality has been developed for a reason. So, although it may not be required in the short-term, it may prove beneficial to you in the longer term. It will always be much easier to "switch it on" rather than develop it internally.



So are you still deciding whether to build or buy?

Conclusion

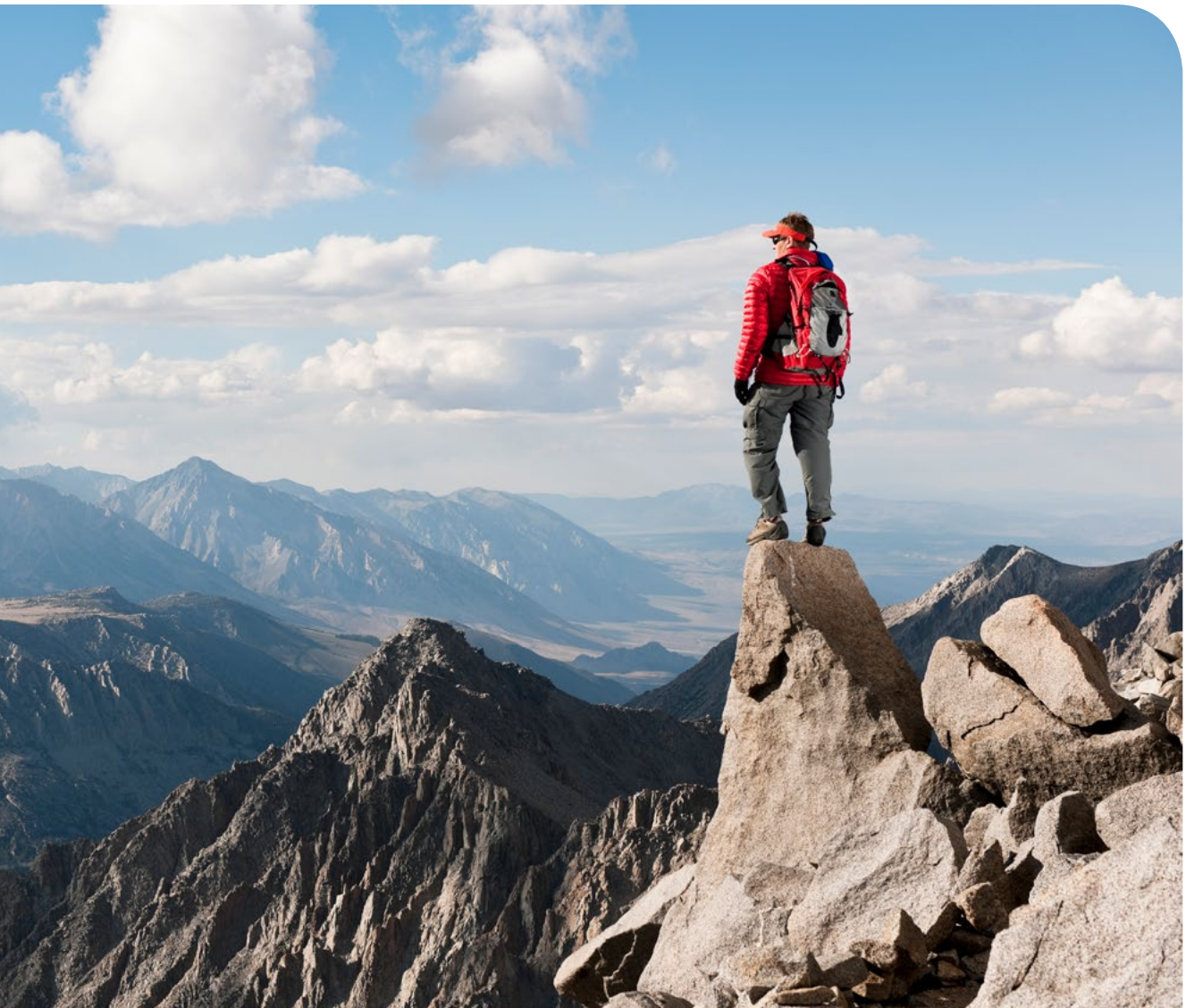
If you've got the skills, budget and time, then like building a house, building your own software application is entirely possible. But it is definitely a far more resource and time intensive exercise that may require more effort long term than most originally plan or expect.

After assessing the pros and cons, my recommendation is that you should build software internally only when a) there is no viable option on the market today or b) the problem can be easily solved.

Although there will generally be some level of customization involved, buying an outsourced software solution generally reduces stress, unplanned expenditure and time associated with building internally as well as offering a faster ROI and a highly scalable solution.

As a final note, remember that if the product already exists in the market, it's usually because there is an unmet need and that the best people to help you deal with it are the ones who work on it every day.

If you've already embarked on the build solution, and are finding it more challenging and less rewarding than you expected, never be afraid to "fold 'em" and walk away. The cost of buying a solution at this stage could still be a lot less than the cost of finalising a never-ending project with a risky outcome.



About iasset.com®

iasset.com® is the leader in revolutionizing global IT channel efficiency. Our cloud-based platform helps reduce complexity and increase revenue for our customers each day. *iasset.com*® empowers the entire IT channel ecosystem – from vendors, distributors/aggregators, to service providers and value-added resellers. We accomplish this by automating the product lifecycle for any type of product or service – including cloud consumption and subscription contracts, and hardware or software maintenance renewal contracts. As a result, our customers achieve superior business outcomes including faster and more streamlined processes, compliance, added intelligence, cost savings and the ability to service their own channel and customers more effectively.

Today, *iasset.com*® manages over $20B worth of assets in more than 150 countries, and continues to be the solution of choice for leading technology organizations.

Want to learn more? Contact us at info@iasset.com or visit www.iasset.com

AMERICAS
Tel: +1 415 745 3568

ASIA PACIFIC
Tel: +61 2 8915 6222

EUROPE
Tel: 0800 048 8970

iasset com®
The Global Channel Ecosystem