Buy Versus Build and Solution Selection Analysis

By Craig Fisher, CEO

Pango Technology

Pango Technology, Inc.
502 E. Fireweed Lane
Anchorage, AK 99503-2844

907-868-8092
pangoinfo@pangotechnology.com
www.pangotechnology.com


PANGO
TECHNOLOGY

When faced with a new software or technology need, businesses often encounter the question:

Should we buy an off-the-shelf solution, or should we build something custom to fit our unique needs?

or more broadly,

How do I pick the right software to meet my need?

A Buy Versus Build/Solution Selection (Buy/Build) analysis is a common business assessment that helps companies decide between purchasing one of the commercially available pre-made solutions or developing a custom solution tailored to their requirements. Buy/Build analysis is a form of Solution Selection Analysis where you explicitly include a custom build option when you are considering possible solutions.

Whether you know it or not, every time you buy software you perform a Buy/Build analysis. It is just a question of how much time and attention you give it. A well-executed Buy/Build analysis is critical. The wrong software can damage company growth, resource allocation, and competitive advantage. Paying close attention to that decision only makes sense.

A thorough Buy/Build analysis helps minimize risk and maximize return on investment (ROI). With a careful review of options, companies can avoid costly mistakes, such as:

1. Selecting a solution that does not fully meet their needs.
2. Underestimating the time and resources required for implementation.
3. Delayed timelines.
4. Misaligned solutions that create more problems than they solve.

Certain situations commonly trigger the need for new software. These can include:

- **New business needs**, such as launching a new product or expanding to new markets.
- **Upgrades or replacements** for outdated systems.
- **Scalability requirements** to support growth.

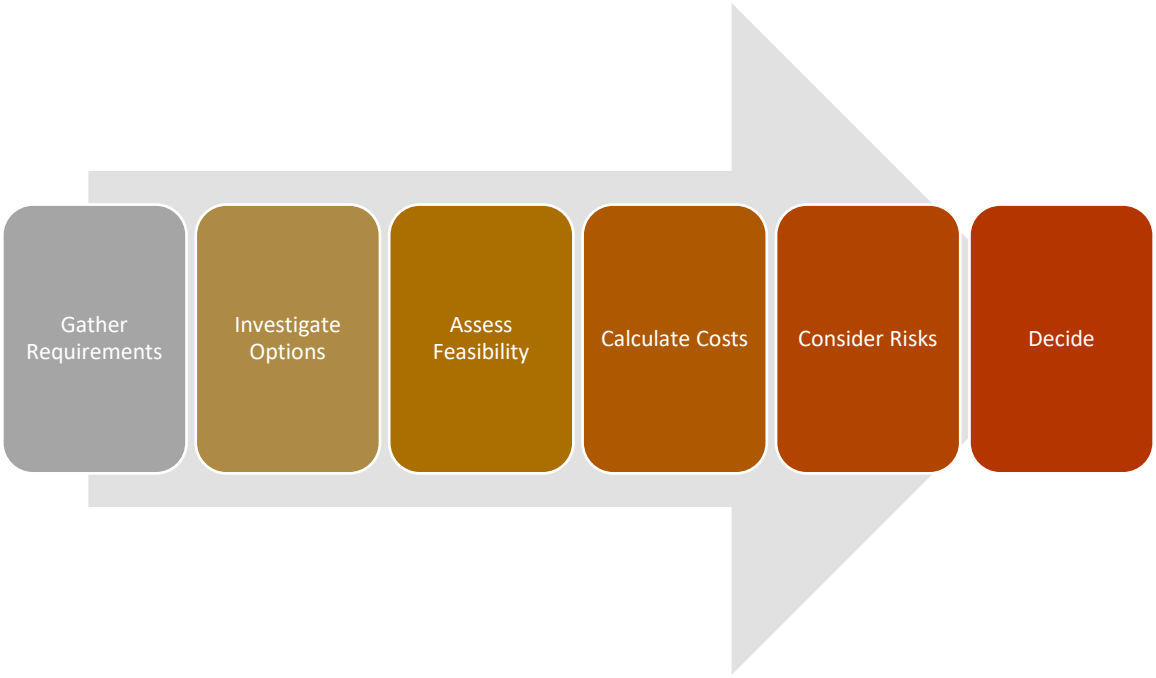
Cost, time, long-term goals, and available resources are examples of key factors that drive this analysis.

This article provides a straightforward look at how to conduct a Buy/Build analysis. We will cover the basics, outline the process, discuss common pitfalls, to help you gain the knowledge needed to make an informed decision for your business.

If, at this point, you find yourself thinking that these decisions do not seem all that complicated, you may be right! Sometimes the choice is clear even without a lot of analysis. If you need an office suite (word processor, spreadsheet, etc..), you are probably going to select between the 2 market leaders based on what the people in your organization already know and use. You definitely should not write your own! Nonetheless, understanding the steps in a formal Buy/build or Solution Selection analysis may help you identify the times where more caution and rigor is advised.

The Basic Process

The basic process for executing a Buy/Build analysis looks like this:



Step 1: Gather Requirements

Software requirements capture what the system must do to meet your business needs, and getting this right from the start can save time, money, and frustration down the line.

Start by identifying and involving all the stakeholder groups. Stakeholders include anyone who uses, benefits from, or is affected by the software. This might include employees, managers, customers, or partners. Talking to people from these groups will help you understand their challenges, goals, and expectations for the solution. Ask questions like: What problem are we solving? What features are essential? What is wrong with the current solution? What is right about it? What does success look like?

Finding the correct level of specificity for your requirements is challenging at this stage. You need to go deep enough to adequately understand the costs of a custom-build but not so deep that no off-the-shelf software can meet those functional requirements. Focus on business needs and processes rather than software features.

Once you have gathered input, organize and prioritize the requirements. Not all needs are created equal. Some features will be “must-haves” essential to the software’s core function, while others might be “nice-to-haves” that enhance the user experience. Clearly categorizing these can help you focus on important aspects of the solution and ensure you are allocating resources effectively.

It is also important to document everything clearly. Requirements should be specific, measurable, and easy for both technical and non-technical stakeholders to understand. For example, instead of saying,

"The system should improve customer service," specify, "The system should allow customer service representatives to access customer order history within three clicks and respond to inquiries within 30 seconds." This level of detail ensures that expectations are clear, measurable, and aligned with business goals.

It is essential that the requirements process generates consensus with the stakeholder community about what is and is not important and more generally what the new software will and will not do. Gaining consensus should be an active process wherein you engage with stakeholders and actively solicit their reactions to the written requirements. This helps ensure that you have properly captured business needs.

Well-defined requirements provide a roadmap to guide your team toward building—or buying—a solution that truly delivers value.

Step 2: Investigate Options

Now you should develop a list of likely solutions so that you can begin to make your decision. Look around at what other organizations are doing to solve the same problem. Evaluate well-reviewed products with a strong track record, particularly those used by businesses like yours. Vendor websites, industry forums, and peer recommendations can be valuable resources for identifying potential options.

Ask the following questions about each option:

1. **How closely does the solution match my business requirements?**
   This may be the most important question. A tool that solves the wrong problem is not worth much! On the other hand, you may need to change your processes to better match an otherwise good tool. Will this software grow with my business, or will it prove to be a bottleneck to future growth?
2. **Does the proposed software match my organization's technical skills?**
   Software requires care and feeding. Depending on the tool you select and how it is implemented you may need to run updates on the software, its framework, or server. You may also need to monitor performance, support users, develop integrations, add features, etc. If your organization cannot support the software in question, pick something else with a technical architecture that more closely matches your organization's capabilities.[1]
3. **Does the proposed software meet my security requirements?**
   Depending on your industry and the system in question, there are several security standards that may apply to this analysis. Ensure that the selected solution will adhere to those mandated requirements through conversation with the vendors. When thinking more generally about security requirements, consider the data stored and managed by the system. What are the implications if that data gets out? Make sure your potential vendors understand those risks and have taken appropriate measures to mitigate them.
4. **How reliable is the proposed vendor?**
   You are probably going to use the software for a long time and switching vendors costs time and money. Choose software that is backed by a company you can trust. Examine the vendor's track record for reliability. Have conversations with existing customers. Talk with the vendor about

[1] SAAS/Cloud architectures can help alleviate some of these concerns.

their guarantees around service responsiveness. Ask the vendor about their roadmap for the software – they should be able to describe their plans and coming features. Look for reports of outages, breaches, and even financial instability. For custom builds, ensure the development team has experience delivering similar projects and can provide ongoing support.

Assemble a list of likely solutions and the information outlined above. Consider product demonstrations, existing customer interviews, public financial filings, and even YouTube videos while gathering this information. A well-researched list of a few solutions will make the next steps easier and more productive.

Step 3: Assess Feasibility

In this step you will consider the feasibility of each solution and rank them in order from most feasible to least. Which solution most closely meets the business requirements outlined in Step 1? Which solution seems most likely to grow with your business? Look at your resources, available expertise, and timeline to see which solution seems to be the best match for your organization. How well does each solution do with security? How do you rate the reliability of the vendor? Your goal in this phase is to rank these solutions so that you can focus on the most likely contenders. We often assign a score of 1 – 5 to each major question where 1 is a poor fit and 5 is great. Add up the score for each solution and sort from high to low score. The highest scored solutions deserve the most attention.

Step 4: Calculate Costs

Calculating the cost of software implementations is difficult and time consuming. You may want to limit your efforts to a few of the top-ranking solutions.

It is essential to consider not just the upfront costs but also the total cost of ownership (TCO). Upfront costs are often the most visible, they include licensing fees for off-the-shelf products or development costs for custom-built solutions. However, these factors are just the beginning. An actual cost evaluation must account for all the expenses you will incur over the life of the software, giving you a clearer picture of its long-term financial impact. A standard approach to comparing these costs is to calculate a 5-year TCO for each option. There is a lot of estimation (read guesswork) in this process, but the exercise is still useful when comparing options and assessing risk.

For off-the-shelf software, upfront costs usually include license or subscription fees as well as implementation costs. Keep in mind that large-scale software implementations do not happen overnight so you will lose staff time to requirements discussions, training, and outages during the process. Depending on the size and impact of the implementation you may even need to supplement your staff during that period. Beyond these initial expenses, factor in ongoing costs like subscription renewals, technical support, on-going training for employees, and potential customization fees. Some off-the-shelf solutions may also require integration with your existing tools or systems, so be sure to include those in your calculations.

Custom-built solutions typically have higher upfront costs due to design and development. They usually do not have licensing fees, but they still come with ongoing expenses that are important to anticipate. Maintenance, updates, and bug fixes are all part of the cost of ownership. If you do not have an in-house development team, you will need to contract with the original developers or hire specialists for long-term support. While custom solutions offer greater flexibility, they also require consistent investment to

keep them functional and relevant. On the positive side, you should not need to change business processes to conform to the way the system works like you do with off-the-shelf software. That means you can keep doing the important things that make your customers happy!

You should also consider hidden costs. One of the biggest hidden expenses is retooling your process to accommodate the new software. The off-the-shelf software you buy will expect you to change the way you do things to work in the new system. That retooling can be difficult! Also, watch out for functional gaps in the new software. You sometimes discover that the new software is missing some key features and that you must work with the vendor to solve that problem. That may cost more.

By calculating the upfront, ongoing, and hidden costs, you can better understand the total cost of ownership for each option. This comprehensive view helps you make your decision based on what will be sustainable and cost-effective in the long run. Remember, the cheapest option today is not always the most economical choice over time.

Now that you have rough costs for each of the most feasible options you can re-rank your list based on cost.

Step 5: Consider Risks

Consider the risks associated with each ranked option, such as vendor reliability, dependencies on external support, and change management needs. Each of these factors affects long-term stability and adaptability.

It is difficult to measure risk and often requires a judgement call on your part. One approach that we have used is to apply a cost multiplier based on our perception of risk. For example, imagine that the TCO for option 1 is $100,000 over 5 years and for option 2 it is $120,000 over the same period. The problem is that you can't find anyone who has implemented option 1 but you know several people who have been successful with option 2. Option 1 is clearly riskier than option 2. Is it 10% more risky? Or 20%? Or 30%? This is where it gets subjective but for argument's sake, we are going with a risk adjustment of 30% for option 1 and 0% for option 2. Your risk adjusted cost for option 1 is $130,000 and option 2 is still $120,000. Option 2 is probably the better deal.

This approach is necessarily subjective but still informed by strength and weaknesses of each solution gathered and analyzed above.

Step 6: Decide

The final decision should align with your company's strategic goals and operational realities. Choosing the right solution means balancing short-term needs with long-term strategy, ensuring the solution supports overall growth. Work with your team of stakeholders to review the findings of the study as they will have to live with this decision as much as you will.

Dealing with Complexity

Software powers business. It is often at the heart of what you do every day. It can shape your business for better or worse. It is critical to find a solution that works well with your business processes, is reliable, maintainable, affordable, and will grow with you.

Picking the right software is both important and difficult. There are many factors to consider, and the

situation is often rife with uncertainty. The framework described above will help you untangle the complexity and the uncertainty, but it is a lot of work.

From time-to-time organizations get help with this process. Firms like Pango Technology can help you through this process. We are experts in the process and can help help you find the best possible solution, whether custom or off-the-shelf.

Whether you bring in a team of third-party analysts or you take on the job yourself, the process is likely to be expensive and slower than you would really prefer. There are a few ideas that can speed up the process.

Before you even start with the above analysis consider the following short cuts:

Commodity Capability versus Core Differentiator

If the need you are addressing is a standard, widely used capability, such as payroll processing or email marketing, it is called a *commodity capacity*, and you should probably buy the software to manage that process. If, on the other hand, the solution directly impacts your business's unique value proposition it is a *core differentiator*, and you should consider a custom-built solution.

Imagine a pizza restaurant. What makes a pizza restaurant great? Is it the point-of-sale software that they use when you are paying the bill? Or is it their fantastic crust? Processing a customer's bill is important, even critical, but it is something that every other pizza place does – as such, it is a commodity capability. That crust however… mmm…THAT is a differentiator.

Time to Market Pressure

If you need it fast, you should buy. Custom software takes time. It is not to say that you can snap your fingers, and the new software will be up and running, with everyone trained, and the data conversion complete and accurate. Software implementations take time -- but less than custom builds.

Short Term Budget Constraints

The upfront costs for custom software will be higher than buying the software off the shelf. Custom software pays off over the long term.

Security and Compliance Requirements

If you work in a highly regulated environment, an off-the-self software vendor specializing in compliant, secure solutions might be the best choice, sparing your team the responsibility of meeting these standards from scratch.

Risk Tolerance

Buying can reduce technical risks, as established products have been tested in various environments. Building allows for more customization but may introduce new challenges. Custom built software often carries more schedule, budget, and security risks. There are still risks associated with off-the-shelf software, but they are lower and consequences often less severe.

Short Term Need

Buying a pre-built solution is often a better choice if the need is only temporary. The higher up front cost of custom software is only ever justified in the long term.

Conclusions

The deeper truth here is that buying off-the-shelf software is the right decision in most circumstances. This was not always the case but as more and more software automates more and more business processes and as that software becomes more and more commodified, it just makes sense. That does not mean that custom software is never the right answer, but it is increasingly rare.

Another truth is that picking the right solution, custom or not is often hard and time-consuming while also very important and maybe even existential. What a great combo! Getting help with this crucial process is often a very smart move that increases your chance of success.