## Evaluating solution approaches

# Build vs. Buy - A Decision-Making Framework for Software Development

### 7 Key Considerations When Adding Report Functionality to Your Application

[**Download the full White Paper**](https://cdn.windwardstudios.com/Website/WhitePapers/Build-vs-Buy.pdf)

If you’re planning a new application, or in the middle of building one, you should be actively considering your reporting needs and development strategy. If your application is already deployed, you should be evaluating your user’s satisfaction with its reporting capabilities and making plans to raise the bar.

Most applications need some level of reporting functionality, and getting it right reduces operational costs, increase productivity, improve user satisfaction or even provide product differentiation.

So, regardless of the stage or situation, you know that you’ve got important decisions to make that will involve weighing the pros and cons of building or buying your reporting solution.

In this White Paper we’ll look at the major factors affecting your decision including: questions, considerations, options and the advantages to each approach. [Download the full Build vs. Buy PDF](https://assets-global.website-files.com/5a8e038236499300015e031c/5bc8bee0662bc6f627f50bea_Build-vs-Buy.pdf?utm_source=website&utm_content=buildvsbuy) to follow along with the scoring worksheet. Choose whether ‘Build’ or ‘Buy’ wins in each of the 7 Key Considerations, rank their weight in your decision, and use the final score as a recommended course of action.

Keep in mind, there is no right or wrong answer—what is right for one company may not work for the other. And in the end, it will probably depend on how you prioritize these 7 Key Considerations.

Let’s take a look at the major factors and considerations you will face while making your decision.

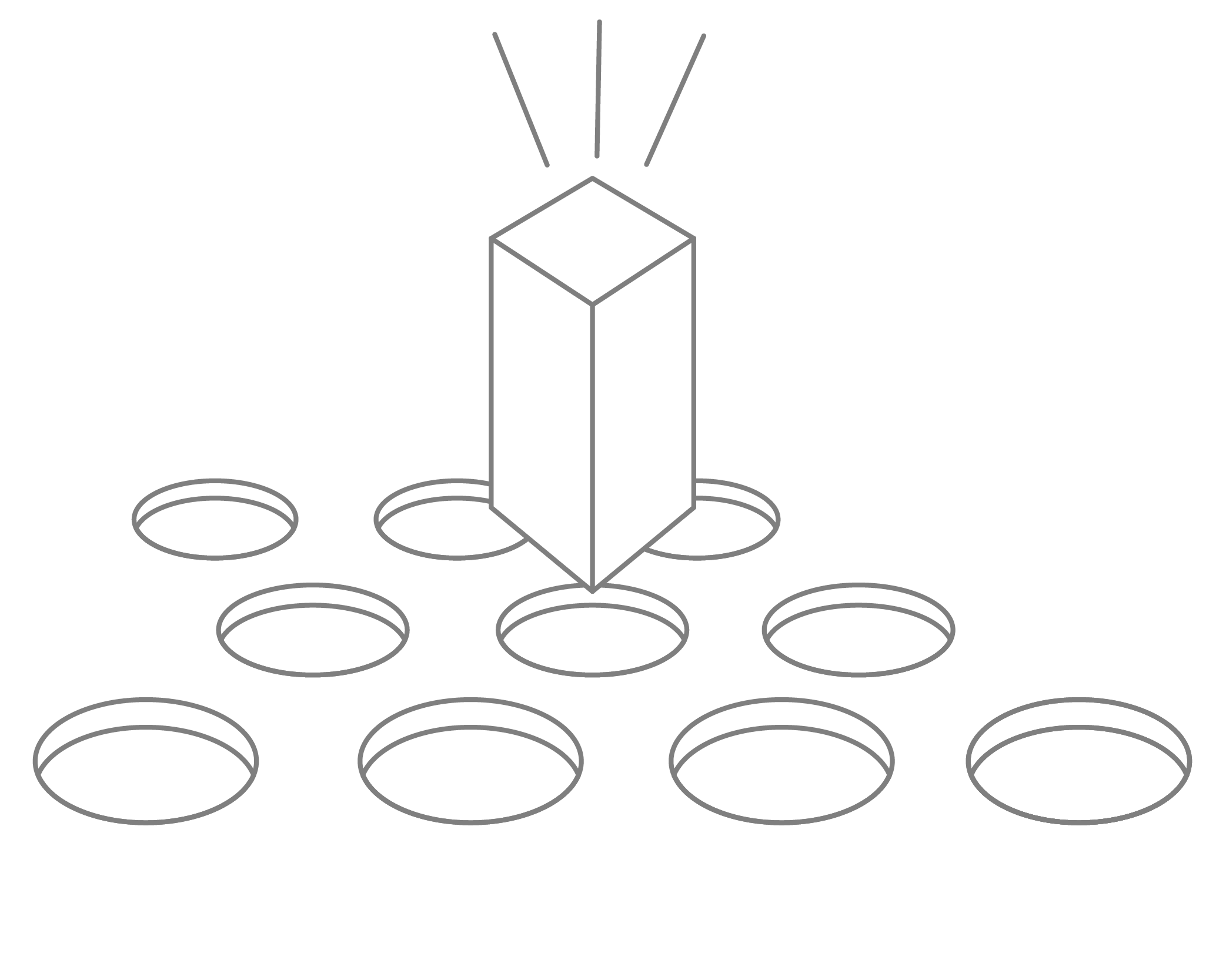
## 1. Technology & Compatibility

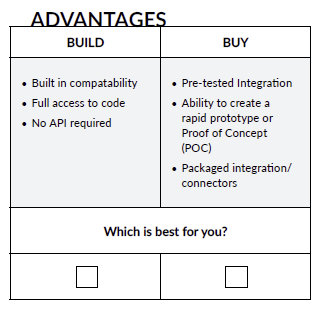
#### Square Pegs & Round Holes

Integration compatibility can be a deal-breaker when deciding on your report solution. Some solutions are just unfit for your application—like the saying “square pegs don’t fit in round holes”.

Among third-party software apps, ease of integration varies widely. However, many packaged software solutions make it possible to perform a proof of concept (POC) before investing in a solution.

If you choose to build your solution, compatibility is not likely to be an issue. However you’ll still need to consider how your solution will connect to it’s potentially disparate data sources.





### ‍

### ****Questions to Ask Yourself:****

##### What is our technology platform?

##### What are our data source(s)? And are they accessible and available?

##### What are our output format requirements (i.e. print, display, PDF)? And how will they change over time?

##### What is the level of integration needed? Fully Integrated? Connected? Stand Alone?

##### What packaged solutions are available? What platforms are they compatible with?

‍

## 2. Features

#### The Must Have’s, Should Have’s, Could Have’s, & Won’t Have’s

Some organizations use the **MoSCoW** method to prioritize feature requirements.

**Must Have**- Critical for success or otherwise failure.

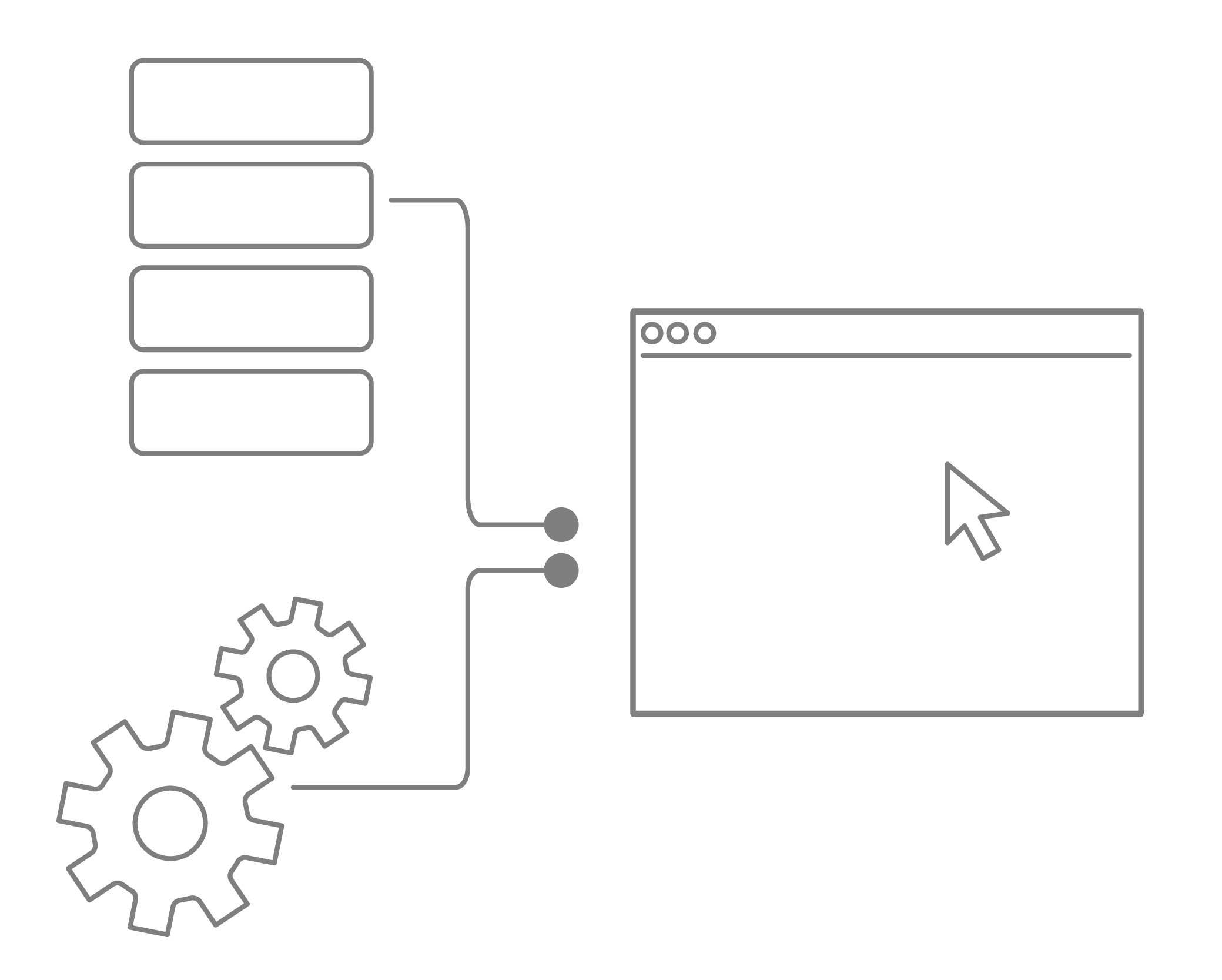
**Should Have**- Important but not necessary or critical.

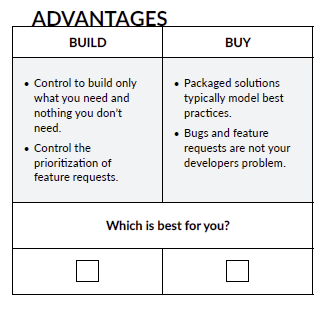
**Could Have**- Desirable but not necessary. Or if time and resources permit.

**Won’t Have**- Least critical or for a later time.

It’s important to understand your end-users’ needs and how they are driving your prioritization—know what is non-negotiable and what is noise.

When considering packaged solutions it’s also important to look beyond what the software claims it can do, to what it does best. For example, report solutions that are strong in BI analytics often have little to no document generation capability. You may find that two separate packages is an ideal way to offer a best-in-class total solution.





### ****Questions to Ask Yourself:****

##### What will the output be used for: executive dashboards or interactive analysis, print or presentation, ad-hoc or scheduled?

##### Do we have clear prioritization from ‘Must Have’ to ‘Won’t Have’?

##### Will feature demands increase with growth of the application or company?

##### As users become more sophisticated, will they need new or better features?

## 3. Performance

#### The Need for Speed

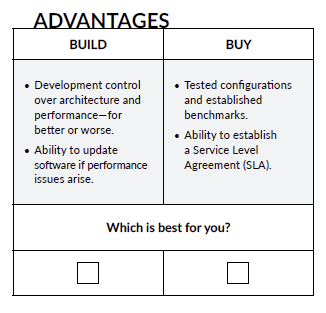
The most elegant solution is only as good as its ability to meet performance requirements.

Not only must you establish your current performance needs, but you must look into the future to determine how your needs may change over time, and how you will address them.

Within any solution, there are countless potential points of failure and performance bottlenecks.

You wouldn’t show up to a Formula One race driving a school bus. You need to be certain your equipment is up to the task.

‍



### ****Questions to Ask Yourself:****

##### What are our output volume requirements: ad hoc, on demand or hundreds-of-thousands of documents each day?

##### How will our needs change over time? Will our customer base change? Will our reporting needs become more complex? Can scale be addressed with hardware, software or both?

##### How will we test, measure and assure performance?

##### If downtime occurs, what is our exposure and risk?

##### Where might our performance bottleneck exist: connection speed, network speed, database performance, processing, etc...?

‍

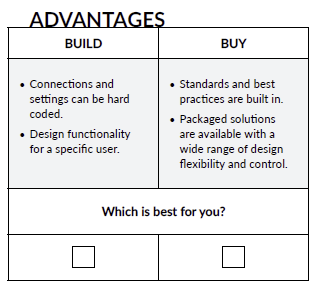
## 4. Design

#### It’s All About the Experience

Will you have a developer designing or a designer developing your reports? If the prior is true you’ll need some beautiful pre-built templates. If the latter is true, you’ll need a very intuitive interface. Of course, having both would be best for everyone.

Any well-designed report is able to communicate the right information quickly and clearly. But customer facing documents have the added responsibility of reinforcing a company’s brand, style and professionalism. You’ll definitely need some amount of design flexibility but in the case of branded documents you’ll likely need very fine free-form design control.

You can learn more about creating better reports, in our White Paper: Design Tips for Beautiful Reports.



### ****Questions to Ask Yourself:****

##### Who are our users: customers, developers, internal business users?

##### Will the designer and developer be working closely together or will there be multiple designers working more autonomously?

##### How important is the user experience?

##### Are canned report templates sufficient, or will our users need design capabilities?

##### How much design flexibility and control are needed?

##### How often will report documents be created and revised?

‍

## 5. Documentation, Training & Support

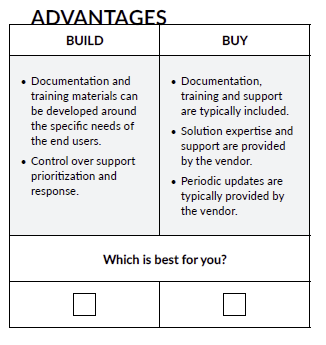
#### Forget Something?

The success of your reporting solution depends on your user’s ability to learn the tools and get the results they want.

Reporting solutions tend to be complex. At least at some level, working with data can be confusing or tedious even for experiencd analysts. Most rely on a collection of tools to help reduce errors and increase their productivity, and each of those tools is likely to approach things a little differently and have unique user interfaces.

Documentation, training and support can make a huge difference in the success of your reporting solution. Unfortunately, they are often an afterthought, or under-developed components of the complete solution. Developing materials and providing ongoing support can easily consume as much, or more, resources than writing the original code.

You will want to carefully weigh the need and effort as you’re making your [build/buy decision](https://www.windwardstudios.com/blog/build-buy-framework-development).



### ****Questions to Ask Yourself:****

##### How intuitive and how complex is the solution we’ve designed or purchased? Will we need in-depth documentation, training, or just “quick-start” level material?

##### How often will documentation need to be updated due to feature and version changes?

##### How will ongoing support be handled? And who will train the support team?

##### What is the initial cost of material development? And what is the cost of ongoing support?

‍

## 6. Speed to Market

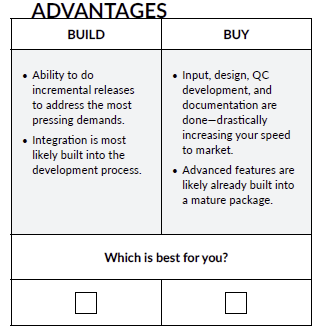
#### Forget Something?

The success of your reporting solution depends on your user’s ability to learn the tools and get the results they want.

Reporting solutions tend to be complex. At least at some level, working with data can be confusing or tedious even for experienced analysts. Most rely on a collection of tools to help reduce errors and increase their productivity, and each of those tools is likely to approach things a little differently and have unique user interfaces.

Documentation, training and support can make a huge difference in the success of your reporting solution. Unfortunately, they are often an afterthought, or under-developed components of the complete solution. Developing materials and providing ongoing support can easily consume as much, or more, resources than writing the original code.

You will want to carefully weigh the need and effort as you’re making your build/buy decision.



### ****Questions to Ask Yourself:****

##### What or who is driving the timeline?

##### What are the potential consequences of delivering later versus sooner?

##### Have we considered all of the components of a complete delivered solution in the project estimate: input, design, POC, development, integration, QC, documentation, training, etc.?

##### What factors may affect our ability to deliver a solution on a timely schedule? To what degree do we have control over those factors?

‍

## 7. Cost & ROI

#### Justification

ROI is the bottom-line look at your ‘Build’ vs. ‘Buy’ decision. The beauty of an ROI analysis is that you’ll have a black-and-white number to look at and share with other stakeholders. The downside is that the number will only be as good as your assumptions and thoroughness. This document should help identify the primary costs and payback opportunities. But you should definitely do some digging to uncover the less obvious factors that may be unique to your organization.

One of the biggest variables is estimating development costs. There are volumes written on how to do this, yet few would claim that it can be done with certainty. Most would agree that various sizing, scoping, prototyping or POC exercises will improve the accuracy of the estimates but those can add significant cost themselves.

It’s important to have a realistic view of the margin of error in your build estimates and your level of confidence in a potential vendor’s ability to deliver as promised.

## Implementing commercial software

## COTS and MOTS

# Modified Off-The-Shelf (MOTS)

## What Does Modified Off-The-Shelf (MOTS) Mean?

Modified off-the- Shelf (MOTS) is a type of software solution that can be modified and customized after being purchased from the software vendor. MOTS is a software delivery concept that enables source code or programmatic customization of a standard prepackaged, market-available software.

## Techopedia Explains Modified Off-The-Shelf (MOTS)

MOTS is designed to be used for organizations that prefer predeveloped software, which can be slightly or substantially customized to meet business objectives. MOTS-based software solutions provide partial or complete access to the source code of the underlying software. The software buyer can review the code and product literature to modify the appearance, functionality and/or the business logic of the software. Moreover, the software vendor, whether directly or through a development/integration partner, executes and manages the software modification/customization. MOTS is the opposite to COTS, which provides the same commercial software but doesn’t support any code-level modification.

## What Does Commercial Off-The-Shelf (COTS) Mean?

Commercial off-the-shelf (COTS) is a term that references non-developmental items (NDI) sold in the commercial marketplace and used or obtained through government contracts. The set of rules for COTS is defined by the Federal Acquisition Regulation (FAR).

A COTS product is usually a computer hardware or software product tailored for specific uses and made available to the general public. Such products are designed to be readily available and user friendly. A typical example of a COTS product is Microsoft Office or antivirus software. A COTS product is generally any product available off-the-shelf and not requiring custom development before installation.

## Techopedia Explains Commercial Off-The-Shelf (COTS)

Compared to COTS, a custom designed product is typically more expensive and not as dependable. This is because the product is industrialized from scratch in minimal time with a limited budget. COTS that is modified by a purchaser, vendor or other party to meet customer requirements become modified off-the-shelf (MOTS). Generally once a COT is modified, it is the responsibility of the consumer to manage changes to the product.

Procuring COTS products has become a necessity for several big businesses. It is typical for a large organization to incorporate various COTS products into its system for better functionality, as well as being a relatively risk-free investment. This has shaped a larger market for COTS products.