osmos

Should You Partner or Build an Internal Data Importer?

Table of Contents

Introduction

As data exchange between companies grows, your company needs scalable solutions to ingest data from your customers, partners, and vendors. You're looking for ways to safely and securely ingest external data into your operational systems.

That's only half the battle. External data is messy. You have no control over the quality and format of the data incoming from customers and partners. It's compounding and frustrating for your team due to the increasingly manual and tedious process.

You're feeling the frustration, so it's time to find a fix–and fast. At this point, you're probably aware of some of the solutions on the market or you've considered creating your own bespoke tool. Building your own external ingestion solution for this complex and strategic problem can be beneficial depending on your use case. Beyond the flexibility and configurability of a fully customized tool, you get to control the prioritization of new features and functions.

If you need a custom connector or integration, you won't need to wait to start ingesting data into your system. Plus, when you go the DIY route, there aren't any immediate, upfront software costs, which your finance team may thank you for. But before you jump headfirst into building your own solution, we've outlined some of the important variables to consider.

Getting Started

1

Goal

Determine your organization’s data ingestion needs.

2

Resource

Estimate the costs and headcount needed to build your solution.

3

Plan

Make a plan for how this project will fit into your roadmap.

4

Time

Estimate the time needed to build, test, launch, and maintain your solution.

5

Launch

Consider the long-term efforts needed for maintenance and support.

11 Things to Consider Before Building an Internal Data Importer

	Build	Buy
Build Time	6-12 months	1 day
Number of Devs	2-3	<1
Complexity	A headache	Out-of-the-box solutions
Sacrificed Roadmap	Yes	No
Go-Live Time	6-12 months	Days
Maintenance	Ongoing	Don’t sweat it
Support Costs	Unpredictable	Included
Scalability	More customers + uses cases = more time	Built to scale
User-Friendliness	Requires extensive user research	Built with your end-users in mind
Cost	One dev’s salary x number of devs needed x time to build + ongoing maintenance costs	As low as $299/month
Opportunity Cost	Falling behind your competition	None

1. Developer Time

Building your own data ingestion tool can be an extremely time intensive endeavor for your technical team. Whether you need a self-serve data uploader or an ETL pipeline, the timeline is dependent on the complexity and number of resources needed. For the simplest data importer, it could take a dev team 1-3 weeks to build. But that's for the most basic of data ingestion tools.

If your incoming data requires significant validations and cleaning, expect between 6 to 12 months of dev time devoted to this project. The more complex the transformations, the more time your team will need to build a data cleansing engine. You can count on needing extra time if you want additional connectors or integrations, too.

"My engineering focus now goes towards core product improvements and bringing peace of mind to more customers."

Nishant Mathew,
CTO of Spot

2. Number of Developers Involved

The state of the job market recently has shown us the importance of hiring–and retaining the right talent. This is especially true of developers. In fact, 60% of CIOs have complained that the shortage of technical talent has impacted their ability to keep pace with their top competitors.

It is more important than ever to properly allocate your teams' skills to build and refine your own product. Add building homegrown data pipelines to the roadmap, and your talent's focus is shifted away from impactful, revenue generating projects.

3. Complexities

At this point you might be thinking, “well how hard can it be?” So let’s break it down. Initially, ingesting external data seems like a simple problem to solve. But it’s important to consider the complexity beforehand, so this process doesn’t take up extra dev cycles. Complex features like appropriately detailed logging (to enable breakfix) and ML automapping help save dev time in the long run and improve the customer experience, but are a lot of work up front.

External data is inherently messy since there is no universal standard for sharing data. This adds a layer of complexity that requires sophisticated software. You need to have validations within your data ingestion solution to handle multiple scenarios, so any errors are caught before they pollute the destination. However, validation failures need to be handled in a way that won’t bring your process to a grinding halt every time there is an error or when your devs need to deploy a fix.

Adding to the complexity of building your own tool are connectors and integrations. Every system that you need to pull data from or send data to requires a considerable amount of testing. Testing these connectors requires dummy accounts which cost time and money. Plus, it will take dev time to build UI automation to verify that your tests were successful.

4. Sacrificed Product Roadmap

Sacrificing your roadmap for additional projects means new features and product updates fall by the wayside. You need to successfully bring a product to market that satisfies customer needs and the larger business objectives, which is a pretty tall task in and of itself.

So how does building an internal data importing tool fit into that? There are inherent trade-offs that have to be made.

1. First, determine if a DIY data importer will fulfill your long-term vision for your product. If it does, you need to think of how it fits into your current strategy by detailing all of the ways it will fulfill your goals and OKRs.
2. Then, it'll be necessary to outline the product specs which are seemingly endless for a MVP.
3. After that's all figured out, it's time to prioritize. What new features or updates might get pushed to the next sprint, next month, or next quarter?

5. Go-Live Time

Building a custom data ingestion solution is no small feat.

Osmos customer, Mosaic, determined that it would take 9-12 months to build an internal data uploader. And it's not just a simple importer that you're building. Consider the time it takes to create custom validations, connectors and APIs, and automation. Up to a year (or more) of engineering time is nothing to sneeze at.

After a few months of work, Mosaic's CTO, Nima Tayebi, finally decided that they "want to build the best resource management system possible, not the best import tool."

"We want to build the best resource management system possible, not the best data import tool."

Nima Tayebi,
CTO of Mosaic

6. Maintenance

Once you have your internal tool up and running, you'll need to consider the upkeep it takes to maintain data pipelines. Unfortunately, you can't just set it and forget it. Maintaining secure ETL data pipelines requires frequent troubleshooting.

Your team may need to prioritize fixing bugs if the need arises. This can be an overwhelming and unproductive task for your star team to focus their efforts. Plus, it will be important to be proactive as to not incur any long-term problems.

7. Support Costs

Thinking outside of the engineering team, an internal data importer will also affect your GTM teams as well. Customer success (CS), who helps with onboarding customers, is often a key partner for your users. If they're spending lots of time helping customers with data uploads and cleaning messy data, it means less time dedicated to showing your customers the value of your product.

Having intuitive data imports that allow customers to easily serve themselves seriously lowers support costs, while allowing your CS teams to focus on what they're best at–helping your customers with your product.

8. Scalability

As your organization grows, new obstacles will, unfortunately, arise. It is important to make sure any data ingestion solution is able to grow with you and your needs. Building an internal solution can be limiting due to the extensive time and energy needed to build and maintain a custom tool.

Partnering with a provider, however, can offer increased agility, as they often have prebuilt solutions for a variety of problems. It's best to find a platform with end-to-end solutions that can handle multiple use cases. Find a solution provider who is willing to partner with you and offers a product that is fully customizable and configurable to your needs.

"Osmos is a strategic enabler of our global platform, making it easier to bring solutions to our customers, partners, and distributors."

Matt Robinson,
CTO of Rahi

9. User-Friendliness

We all know the amount of research that goes into making a user-friendly product with an intuitive UI. If you plan to build an internal data import tool, it needs to be simple enough that your nontechnical internal teams and customers or partners can quickly and easily upload data without frustration.

Leave it to the experts who’ve done extensive user research to build a simplistic and easy-to-use interface that enables one-click uploads and empowers end-users to upload their clean data directly into your target systems.

10. Cost

While evaluating your options, cost is an important factor for your organization to take into account. Not only does it come down to money, but also the amount of time and resources needed to solve your problem. Building the most simple data pipeline could take weeks to months and that doesn't include AI-powered data transformations, custom validations and business rules, or additional connectors and integrations.

If it takes multiple devs or even hiring additional headcount to build an internal tool, then their time and compensation need to be factored in. The cost to build it yourself will be significantly more than even the most complex and expensive solutions on the market.

While you have your calculator handy, it's worth considering ongoing costs to maintain a bespoke tool. Any kind of dev support and upkeep will of course cost extra dollars. But it is also important to factor in the cost of adding new connectors and new features which can seem like a never ending cycle as your organization continues to scale.

11. Opportunity Cost

Everyone knows that time is money. And when it comes to building a product, time is one of your most valuable resources. Sacrificing your roadmap to additional projects is now less time than your competitors to release new features and updates.

Falling behind the competition because you and your team shifted focus away from customer needs to internal processes is not a favorable position to be in. An external tool is far faster to implement with so much added functionality, you'll be able to focus on staying ahead in your market.

Quantifying Your Decision

You now have a very important decision to make for your team, so let's take a look to see what factors are most important to your organization. "Build" might win some points while "Buy" might win in other areas.

Take a look at our scorecard on the next page and start by placing a check mark if you prefer to build or buy, then rank the importance of that category on a scale of 1-5 (5 being the most important). Once you've done that, tally up the score for each column to help inform your decision.

	Build	Buy
Build Time	□ 1 2 3 4 5	□ 1 2 3 4 5
Number of Devs	□ 1 2 3 4 5	□ 1 2 3 4 5
Complexity	□ 1 2 3 4 5	□ 1 2 3 4 5
Sacrificed Roadmap	□ 1 2 3 4 5	□ 1 2 3 4 5
Go-Live Time	□ 1 2 3 4 5	□ 1 2 3 4 5
Maintenance	□ 1 2 3 4 5	□ 1 2 3 4 5
Support Costs	□ 1 2 3 4 5	□ 1 2 3 4 5
Scalability	□ 1 2 3 4 5	□ 1 2 3 4 5
User-Friendliness	□ 1 2 3 4 5	□ 1 2 3 4 5
Cost	□ 1 2 3 4 5	□ 1 2 3 4 5
Opportunity Cost	□ 1 2 3 4 5	□ 1 2 3 4 5
Tally each column	Score: ________	Score: ________

Partnering with Osmos

Building your own internal tool requires a slow and costly process before you'll be able to reap the benefits. Partnering with a solution provider will allow you to move faster with less resources.

The Osmos External Data Platform is your product's new growth lever. Our low-code embeddable solutions allow you to get started without a heavy lift from your engineering team. So you can quickly start ingesting clean data with a fully configurable and customizable solution in a day and stop wasting dev cycles.