When to use Logic Apps

8 minutes

Here, we'll discuss how you can decide whether Logic Apps is the right choice for a workflow. We'll list some criteria that indicate whether Logic Apps will meet your performance and functional goals.

Decision criteria

Logic Apps helps you coordinate the flow of data through disparate systems. The cases where Logic Apps might not be the best option typically involve real-time requirements, complex business rules, or use of non-standard services. Here's some discussion of each of these factors.

Integration	The key question to ask when you're considering Logic Apps is *"do I need to integrate services?"* Logic Apps work well when you need to get multiple applications and systems to work together. That's what they were designed to do. If you're building an app with no external connections, Logic Apps is probably not the best option.
Performance	The next consideration is performance. The Logic Apps execution engine scales your apps automatically. Logic Apps can process large data-sets in parallel to let you achieve high throughput. However, they don't guarantee super-fast activation or enforce real-time constraints on execution time. If you're looking for low subsecond response time, then Logic Apps may not be the best fit.
Conditionals	Logic Apps provides control constructs like Boolean expressions, switch statements, and loops so your apps can make decisions based on your data. You can build highly complex and deeply nested conditionals into your Logic Apps. There are two reasons you might prefer not to. First, it's often easier to write conditional logic in code rather than using the Logic Apps Designer. Second, embedded business rules aren't easily sharable with your other apps. Some people like including complex business rules directly in their Logic Apps. Others think it's simpler to write something like an Azure Function to encapsulate the conditional logic and invoke that Function from all their apps.

Connectors	The last consideration is whether there are pre-built connectors for all the services you need to access. If so, then you're ready to go. If not, then you'll need to create a custom connector. If the service has an existing REST or SOAP API, you can make the custom connector in a few hours without writing any code. If not, then you'll need to create the API first before making the connector.

Apply the criteria

Logic Apps works best when you're integrating multiple services with some added control logic. The decision is often a judgment call though. Let's think about how to apply these criteria to our example processes.

Our fictional shoe company needed to monitor social media, move old videos to archive storage, and sell shoes online. Our goal was to decide whether these tasks were good candidates for Logic Apps. To make our decision, we should analyze each task using the four criteria we developed: integration, performance, conditionals, and connectors. The following table summarizes the results. The highlighted cells are discussed below.

	Integration	Performance	Conditionals	Connectors	Use Logic Apps?
Social-media monitor	Integrates multiple services	Doesn't need near-realtime low latency	One simple conditional	Built-in connectors available for all needed systems	Yes
Video archive utility	***Only needs to access one service, cloud storage***	Doesn't need near-realtime low latency	Two simple conditionals	Built-in connectors available for all needed systems	Yes
Direct online sales	Integrates multiple services	Doesn't need near-realtime low latency	***Multiple complex conditionals***	***Multiple custom connectors needed***	***Maybe***

There are a few interesting things to think about in this analysis.

- The video archive task is a good fit for Logic Apps even though it doesn't integrate multiple systems. Logic Apps has a built-in timer trigger and an Azure blob connector that are perfect to implement this process.

- The online sales process would likely include complex business logic. For example, we might have different approval processes based on the purchase amount or different shippers based on the destination. Logic Apps can easily handle these conditions. It's up to us whether we want to embed these business rules in our app.

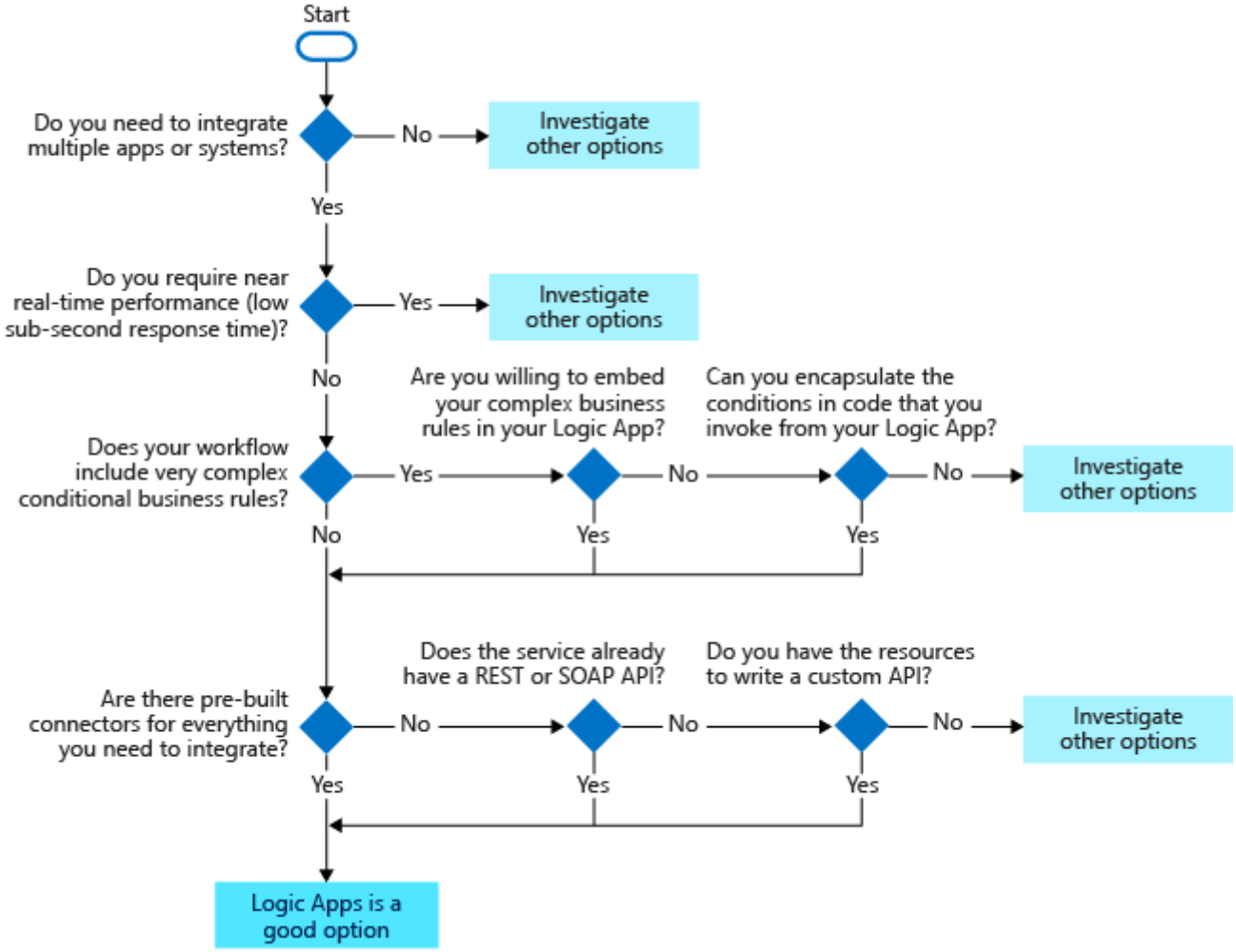
- The online sales process would probably use a mix of built-in and custom connectors. We could use built-in connectors for email notifications and database access but would probably need a custom connector to talk to our payment processing service.

- The performance of Logic Apps will work well for all the tasks. Some of them may process large amounts of data, but Logic Apps scales automatically to handle high throughput or spikes in demand. None of these tasks require very low latency response time. We'd need to have near-realtime constraints for that to be an issue.

Logic Apps could work for all of these tasks. The online sales process is the only one where we'd want to weigh all our options. Logic Apps would be a good choice if we had the resources to build the custom connectors we'd need.

Guidance summary

The following flowchart summarizes the key questions to ask when you're considering using Logic Apps.



Next unit: Knowledge check

Continue >