1. **Choose the best Azure service to automate your business processes**

Objective:

Evaluate Azure services for integration and process automation scenarios

Azure provides several different ways to host and execute code or workflows without using VMs including Azure functions, Microsoft Power Automate, Azure Logic Apps, Azure WebJobs.

**Common business issues.**

In business, one way to guarantee high-quality products and service to customers is to design and implement strict business processes. Such processes may involve multiple steps, people, and software packages.

Problems arise when it merges a second business or integrates with a partner organization. How can administrators integrate the separate processes used in the two organizations, which may have been implemented using different software?

Business processes modeled in software are called **workflows.** Azure includes 4 different technologies that you can use and integrate systems

* Logic apps
* Microsoft power automate
* Web jobs
* Azure functions

They have some similarities:

* They can all accept inputs. An input is a piece of data or a file that is supplied to the workflow
* They can all run actions. An action is a simple operation that the workflow executes and may often modify data or cause another action to be performed.
* They can all include conditions. A condition is a test, often run against an input, that may decide which action to execute next.
* They can all produce outputs. An output is a piece of data or a file that is created by the workflow.

In addition, workflows created with these technologies can either start based on a schedule or they can be triggered by external event.

**Design-first technologies.**

When business analysts discuss and plan business process, they may draw a flow diagram on paper. With logic apps and Microsoft power automate, you can take a similar approach to designing a workflow. They both include user interfaces in which you can draw out the workflow. We call this approach a design first approach.

**Logic apps**

It’s a service within Azure, to automate, orchestrate and integrate disparate components of a distributed application. By using the design-first approach in Logic apps, you can draw out complex workflows that model complex business processes. The following screenshot shows the logic apps designer and design canvas that you use to define your workflow.

Alternatively, you can create or edit a workflow in JSON notation by using the code view.

One reason why Logic apps is so good at integration is that over 200 connectors are included. A connector is a Logic Apps component that provides an interface to an external service. For example. Twitter connector allows you to send and retrieve tweets, office 365 connector lets you manage your email, calendar, contacts. Logic Apps provides hundreds of pre-built connectors that you can use to create your apps. If you have an unusual or unique system that you want to call from logic apps, you can create your own connector if your system exposes a REST API.

**Microsoft Power Automate**

Is a service to create workflows even when you have no development or IT pro experience.

You can create workflows that integrate many different components by using the website or mobile app.

There are four different types of flow you can create:

* **Automated.** Starts by a trigger from some event. For example, the event could be the arrival of a new tweet, or a new file being uploaded.
* **Button.** Runs a repetitive task with a single click from your mobile device.
* **Scheduled.** Runs on a regular basis such like once a week, on a specific date, or after 10 hours
* **Business process.** Models a business process such as the stock ordering process or the complaints procedure. The flow process can have: notification to required people; with their approval recorded; calendar dates for steps, recorded time of flow steps.

Microsoft Power Automate provides an easy-to-use design surface to create flows of the above types.

Under the hood, Microsoft power automate is built on logic apps. Meaning that Power automate supports same range of connectors and actions.

**Design-first technologies compared**

Microsoft Power Automate is more appropriate for use by non-technical staff. If your workflow designers are IT professionals, Logic apps are usually a better fit.

**Code-first technologies**

The developers on your team will likely prefer to write code when they want to orchestrate and integrate different business applications into a single workflow. This is when you need more control over the performance of the workflow or need to write custom code as part of the business process. For such cases, Azure includes Web Jobs and Functions.

**Web Jobs and the Web Jobs SDK**

The Azure app service is cloud based hosting service for web applications, mobile back-ends and RESTful APIs. These applications often need to perform some kind of background task. For example, when photo is uploaded, you may need to generate a smaller thumbnail photograph.

Web Jobs are part of the Azure App Service that you can use to run a program or script automatically. There are two kinds of Web Job:

* **Continuous.** Run in a continuous loop. For example, you could use a continuous Web job to check a shared folder for a new photo.
* **Triggered.** Run when you manually start them or on a schedule.

You can write code in different languages, like PowerShell, Bash. Alternatively, write program in Php, Python, Node.js or Js. Sdk 3.x supports .NET Core.

You can also program a WebJob by using the .NET framework or the .NET Core Framework. In this case, you can use WebJobs SDK to make the task easier. The SDK includes a range of classes, which reduce the amount of code required to interact with the Azure App Service.

WebJobs SDK only supports C# and the NuGet package manager.

**Azure functions**

An azure function is a simple way for you to run small pieces of code in the cloud, without having to worry about the infrastructure required to host that code. You can write Function in C#, Java, JS, Python or any of the languages listed. In addition, you can only pay for the time when the code runs.

When you create an Azure Function, you can start by writing the code for it in the portal. Alternatively, if you need source control, you may use GitHub or Azure DevOps.

To create Azure Function, choose from the range of templates. The following list is a sample of some of the templates available to you:

* HttpTrigger. When you want the code to execute in response to a request sent through the HTTP protocol
* TimeTrigger. When you want the code to execute according to a schedule.
* BlobTrigger. When you want the code to execute when a new blob is added to an Azure Storage account.
* CosmosDbTrigger. When you want the code to execute in response to new or updated documents in a NoSql database.

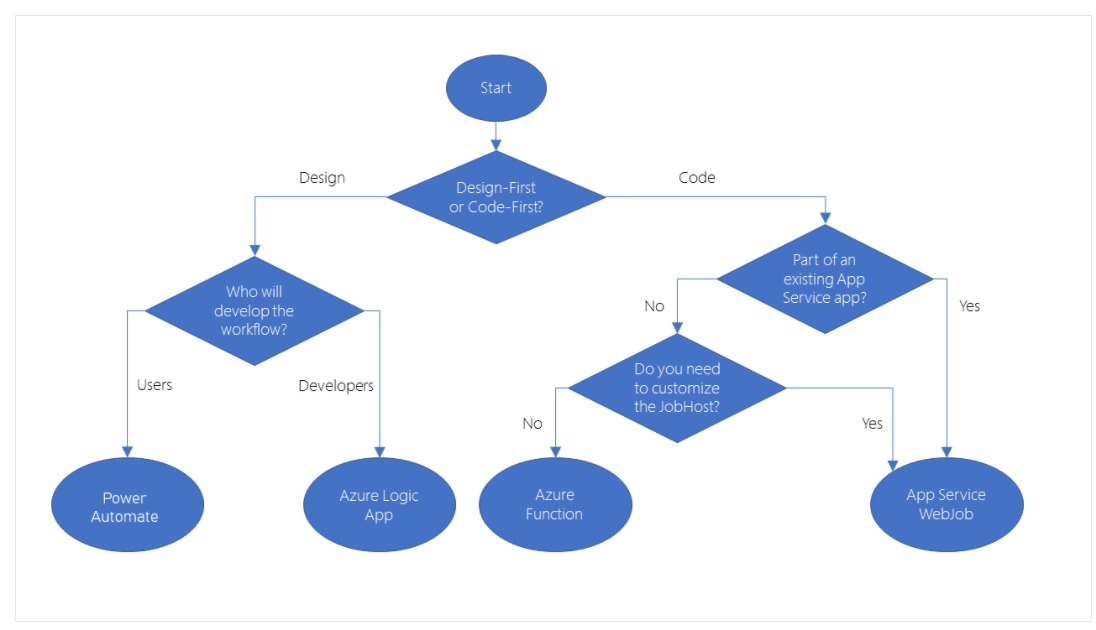
Azure functions can integrate with many different services. These services can trigger your function, or send data input to your function, or receive data output from your function.

**Code-first technologies compared**

In most cases, simple administration and more flexible coding model provided by Azure functions may lead you to chose them in preference to webjobs. However, you may choose WebJobs for the following reasons:

* You want the code to be part of an existing App Service application and to be managed as part of that application, for example in the same Azure DevOps environment.
* You need close control over the object that listens for events that trigger the code.

**Analyze the decision criteria**



First question to ask is if you want GUI or by writing code.

Valid reasons for using a design-first tool:

* People who design the workflow have no coding experience.
* Later designers and users can consult the graphical design to clearly understand how the workflow proceeds

Code-first tool, because:

* People who design the workflow are developers and prefer to work entirely in code
* You want the details of a workflow to be hidden from non-coders.

**Choosing a design-first technology**

The principal question here is who will design the workflow: will it be developers or users?

In logic apps, its designed for people with development skills.

In power automate, its designed for users who have a good understanding of the business process but no coding skills.

**Choosing a code-first technology**

Azure functions should be your default choice (ability to develop test code in the browser, pay per use price model, wider ranges of supported languages and trigger events).

There are situations when WebJobs might be a better choice:

* You have an existing Azure app service application, and you want to model the workflow within the application. This requirement means that the workflow can also be managed as part of the application, for example in Azure DevOps environment.
* You have specific customizations, for example custom retry policy for calls to external systems.

**Mixing technologies**

Remember that there is no requirement for you to use the same technology for different workflows: if your requirements differ, you are likely to reach a different answer at the end of your decision-making process. Furthermore, you can also call one workflow from another. For example, a workflow implemented in Power Automate can easily call another that is built as an Azure Function.

One reason to mix the technologies used in your business processes would be to give users control over a small section of complete workflow.