**12**

**Cost Modeling in the Cloud**

In the previous chapter, we looked extensively at the FinOps processes and how we can implement these in an organization. We also looked at how costs are generated in the various cloud platforms by evaluating the provisioning of resources. Now we have to make sure that costs are allocated in the right way and that they are booked at the right level in the organization.

In this chapter, we will learn how to develop and implement a cost model that allows us to identify cloud costs (showback) and allocate (chargeback) costs to the budgets of teams or units. Before we do that, we must understand the principle of cost coverage, the types of costs in the cloud, and how rates are set by providers.

In this chapter, we’re going to cover the following topics:

* Evaluating the types of cloud costs
* Building a cost model
* Working principles of showback and chargeback

**Evaluating the types of cloud costs**

Before we get to define a cost model, we must understand what types of costs we will be faced with in the cloud. It’s important to have teams understand how costs are being allocated and to have a centralized, controlled, and consistent cost allocation strategy.

**Cost coverage**

A typical cloud deployment model is **pay-as-you-go**(**PAYG**) and that’s OK when the usage of the cloud is limited. Most enterprises will have larger landscapes in the cloud, including critical systems for their business. The company would want to have the guarantee that workloads will always be available, and that capacity is always available. For this, there is the principle of reserving capacity. Reserving capacity—or pre-committing—will bring companies benefits. The first benefit is that they’re sure that capacity is reserved for them and thus available for a longer, contracted period.

The other benefit is that they can have discounts on reserved capacity. The reason for that is that the cloud provider is now ensuring that the customer is committed to using the capacity for a longer period, securing income for the cloud provider. However, there’s also a bit of a downside. Cloud providers can’t freely use the capacity for other customers now, limiting their flexibility in allocating resources in their data centers for those on the PAYG tariff. Hence, cloud providers may charge the customer upfront for reserved capacity. This means that the customer will have to make an investment to secure the capacity. We have to take that into account in our financial and cost models.

For cloud providers, it is all about capacity management. The reserved capacity allows predictability at the data centers. In the current context of global chip shortage affecting many industries, this is very important. But you have the option to pay a reservation monthly and even cancel it (at least in Azure) and apply for other resources/scope or ask for a refund. In case of refunds, additional conditions may apply.

In essence, we will have cloud usage that is covered by reservation and cloud usage that is not. Some usage might not be coverable through reservation. Think of spike usage. A customer could make a reservation for spike usage, but when the resources are removed after the spike has declined, then the reservation would only cost money, without usage. In FinOps this is labeled as **wasted usage**. Resource capacity that is not used but is provisioned will still be charged by the cloud provider. There will likely always be some underutilization of reservation, but then we must make sure that the costs of the underutilization are not exceeding the amount of the discounts. In that case, the reservation will cost the company more than without the discount.

We will try to get to covered usage as much as we can since that will provide the best insights and enable accurate forecasting. Covered usage is a resource that is covered by a reservation, resulting in a lower rate. Now, let’s talk about the rates.

**Cloud rates**

**Rates** are a difficult thing in multi-cloud usage since cloud providers calculate rates in different manners. But there are some guiding principles in understanding how rates are presented to the customer.

We discussed how we can track costs in the cloud. Now, there might be a surprise when the company actually receives the invoice. There might be decreasing rates on the invoice. There might even be different rates calculated for the same type of resource. If a company uses more resources or resources for longer periods, the cloud provider might invoice different rates using, for instance, volume discounts. This can be confusing since the invoice can indeed show various rates for the same type of resources. In that case, FinOps talks about unblended rates.

If there are unblended rates, there will also be blended rates, where costs are evenly distributed across all used resources. Hence, every resource will have the same rate. This might look like a good approach, but it’s not, since blended rates will not tell you exactly what the real costs per resource are.

The rates that eventually show up in the invoice will almost certainly differ from the rates that are shown in the costing portals of the different providers. The public pricing will show the list prices or on-demand rates. Companies hardly pay these full rates. There will be volume discounts, special programs, or otherwise contractually agreed discounts leading to a rate reduction. These reductions can lead to cost savings. The amount of savings will be visible in the invoice and then we can also calculate the savings potential by applying the commercial agreements to our existing and planned cloud environments.

It’s worthwhile to study the various discount programs of cloud providers. In Azure the main ones are:

* **Azure Hybrid Benefit**: This program allows for optimal use of licenses for Windows, SQL Server, Red Hat, and SUSE Linux on virtual machines in Azure (<https://azure.microsoft.com/en-us/pricing/hybrid-benefit/>).
* **Azure Migration and Modernization Program**: This is a program wherein Microsoft offers expert help and best practices to migrate workloads to Azure. Cost optimization and incentives are included to reduce migration costs (<https://azure.microsoft.com/en-us/solutions/migration/migration-modernization-program/#benefits>).

In AWS, an interesting program is **Enterprise Discount Program** (**EDP**). AWS offers enterprise agreements that allow for tailored solutions in AWS. Enterprise discounts for large environments might be applicable. Enterprises that are eligible for this program typically get their own AWS account team that will work closely with the enterprise.

Other clouds have discount programs for big customers too. In most cases, customers will work together with account representatives of the specific cloud provider. In the case of Oracle Cloud, for instance, there are significant benefits when a customer runs Oracle software on top of OCI with software support discounts.

Licenses and agreements influencing rates are a complicated area in the cloud: the advice is really to work with the provider. Rest assured that the provider will try its best to get the best deal for a company that wants to use its cloud. There’s a simple reason for that: without customers, their platforms wouldn’t exist.

**Amortized and fully loaded costs**

We discussed reservations, pre-committing, committing, and discounts. How does that show in costs? We recognize two different sorts of costs here: amortized and fully loaded.

As we have seen in the previous section, reserving capacity might come with upfront investments. In the cost model, we must take these initial upfront costs into account and divide these costs by the actual usage. This is referred to as **amortized** costs.

The **fully loaded** cost is the amortized cost and the actual costs, including the discount rates, plus the shared costs divided by the actual usage. To get to fully loaded costs, we must do a complete breakdown of all cloud components, which we show in the next section.

**Building a cost model**

To build a cost model, we have to follow three basic steps:

1. **Identify cost drivers**: This is something that we discussed in the first two chapters of the book. Cost drivers are closely related to business processes. Think of the number of orders that customers of a company place in a defined time. Identifying the cost drivers will help in setting up the environment. In this particular case, we must think about scaling capacity when ordering intake spikes. But we can also think of event-driven architecture, where the placement of an order triggers a number of events, including invoicing and payment, fetching the order in a warehouse, and distributing the product for delivery. This will likely involve various systems and applications that must communicate with each other. An important question would be what could be automated and how to create the optimal solution since this will have an impact on the costs.
   * One thing that we haven’t touched on yet is the fact that costs will largely define the price that a company can ask for a product or a service. The price of that product is basically cost and margin. A company that simply covers the cost will not make a profit. But the company can also not put any price on their product: it needs to “make sense” to a customer. Anyway, having very detailed, accurate insights into the costs for a company and how it can influence these costs is essential.
   * **Define a full breakdown of all cloud components**: We simply need to know all elements that are used in our clouds. But at the same time, this must be presented in a comprehensible manner. That’s what a cost model does. All expenses must be identified: infrastructure, software, development, security, and so on.
   * **Quantify the components**: We must also answer the question of how much of the specific components we will use. As we have seen in the previous section, is not as simple as **P\*Q** (**price times quantity**) or, in this case, **UC\*Q** (**unit cost times quantity**). We have to take amortized costs into account and various discounts to get to the right levels of cost. In this phase, we should also consider what shared cloud services we could use. Shared services will lower costs, but be aware that there might be a trade-off in compliance.

Let’s start with the breakdown. A best practice is to choose the perspective of the application, assuming that every application will have an application owner. From the application, we can start defining what resources are needed to run that application. There will be unique resources and shared resources. In fact, there will be costs for landing zones that have to be managed. We can have a basic split in costs:

* **Consumption-based**: Everything that responds to the usage of the application.
* **Fixed cost**: Costs that will be made even when there are no applications hosted on the infrastructure. This must be shared among all applications.

Next, we have to think about managed and unmanaged services:

* **Managed**: Services that are completely managed by the provider. There’s no need to add extra costs for activities that a customer must execute themselves to operate the service.
* **Unmanaged**: The cloud provides the service, but the customer has to manage the service itself. This means that there will be additional costing that must be considered. Think of labor, but there might be other costs involved, such as licenses.

This will result in a breakdown per service. In the table below, we list all components of IaaS that will result in a cost. First, the components of the virtual machine. In this example, we used EC2, the service from AWS, but it works exactly the same for virtual machines in any other cloud.

|  |
| --- |
| * EC2 |
| * General |
| * Operating Systems |
| * Licenses |
| * Extensions |
| * RDP Settings |
| * Serial Console |
| * Availability |
| * Virtual Machine SLAs |
| * VM Types |
| * Auto-Shutdown |
| * Just-in-Time Access |
| * Boot Diagnostic |
| * Virtual Server Templates and Infrastructure as Code |
| * Virtual Machine Sizes and Tiers |
| * Virtual Machine Naming Convention |
| * Virtual Machine Deployment and Image Management |
| * Golden Image |
| * Post-Deployment Configuration |
| * Monitoring |
| * AWS EBS |
| * Disk Caching |
| * Drive Type |
| * Drive Performance |
| * EC2 Instance Additional Storage |
| * Managed Drives Reserved Instance |
| * Disk Bursting |
| * Ephemeral OS Disks |
| * Shared Disks |
| * Network Connectivity |
| * Disk Naming Convention |
| * Disk Encryption |

*Table 12.1: Example of a list of components for IaaS*

This is only the VM for the workload. We also need connectivity and storage for that workload. Last but not least, we have to make sure that the workload is secure. The list of components regarding these topics is shown in *Table 12.2*.

|  |
| --- |
| * Network |
| * Private IP Addresses |
| * Public IP Addresses |
| * Accelerated Network |
| * Proximity placement group |
| * Security |
| * Virtual Machine Encryption |
| * Virus Scanning and Anti-Malware |
| * Operating System Update Management |
| * Software Inventory |
| * Host Firewall |
| * Operating Systems Editions |
| * OS Hardening |
| * Storage Accounts |
| * Storage Accounts Security |
| * Authorization |
| * Encryption |
| * Recovery |
| * Tracking |
| * Backup and Recovery |
| * Cloud-Native Backup |
| * AWS Backups |
| * Windows and Linux Server Backup |
| * RDS Backup |
| * File-Based Restore |
| * Backup Monitoring |

*Table 12.2: Example of a list of components for securing a workload*

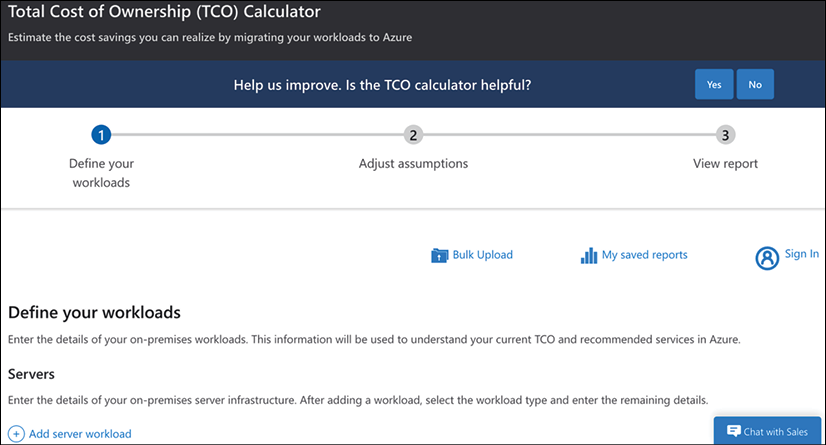
All these components have a cost associated with them. All these costs must be tracked and managed. Hence, we also need to specify cost management settings. *Table 12.3*shows the activities for cost management.

|  |
| --- |
| * Cost Management |
| * Cost Visibility |
| * Cost Reporting and Analysis |
| * Management of Invoices and Payments |
| * Setting and Management of Budgets |
| * Control Usage via Policy |
| * Cost Optimization |
| * Reserved Instance |
| * Spot Instances |
| * Serverless |
| * Trusted Advisor |

*Table 12.3: Components for cost management*

This is an extensive exercise but is needed to create a cost-aware design. You’ve guessed it: this must be done for every artifact in the cloud landscape, for every cloud that we use.

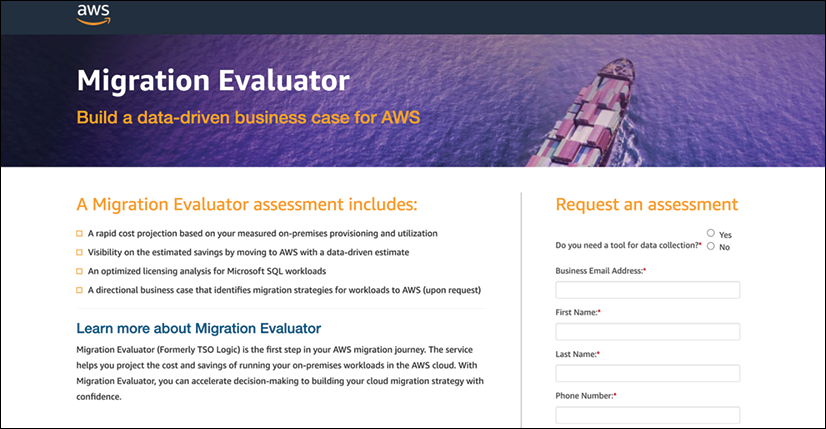
The second step is defining the quantities to start calculating the **total cost of ownership** (**TCO**). In Azure, we can use the TCO calculator for this. There are two ways of working with this calculator. We can start defining our workloads one by one. In that case, we visit the calculator at <https://azure.microsoft.com/en-us/pricing/tco/calculator/>. This is the screen that will be presented:



*Figure 12.1: Starting screen of the Azure TCO calculator*

At the bottom of the screen, we see that we can add a server workload. If we scroll further down, the calculator will present the basic services that we need to host workloads in Azure: databases, storage, and networking. At the top of the screen, we will find the option to do a “bulk upload,” saving a lot of time. Next, we can adjust assumptions and view the report, providing us with the expected TCO when we move workloads to Azure.

Of course, other cloud providers have similar services. Next to pricing calculators, AWS offers Migration Evaluator (<https://aws.amazon.com/migration-evaluator/>). However, this has a different approach. This is an assessment that needs to be requested, as shown in the screenshot:



*Figure 12.2: Starting the request for Migration Evaluator in AWS*

Next, AWS will ask us to upload data, either through, for instance, spreadsheets or automatically collected by installing an agentless collector on existing systems that gathers the data. Migration Evaluator next analyzes the data, estimating the costs and savings in AWS. So, this tool calculates the business case when workloads are migrated to AWS. In Azure, this can be compared to Azure Migrate.

Last, Google Cloud, like AWS, offers an assessment of cloud migration and modernization costs. This assessment can be requested through <https://inthecloud.withgoogle.com/tco-assessment-19/form.html>.

We’ve built the cost model and also assessed the total cost of ownership. The final step is allocation. **Allocation** will help to determine if teams, business units, or departments that are accountable for a product or a service are profitable or not. Cost allocation will reveal cost utilization and help management make decisions. Hence, we must define a logical way to implement cost allocation, which is not easy in the cloud and especially not in multi-cloud. We must map the utilized cloud services to business artifacts or functions to get a clear view of what services are used by who, for what, and what the spend is. A way to do this is by taking the perspective of the application or application group. We will elaborate on this in the next section.

We have now prepared a cost model and we can start monitoring our cloud spend. These costs must be allocated to groups that we have defined. This can be done by showback and chargeback, the topic of the final section of this chapter.

**Working principles of showback and chargeback**

In the previous sections, we learned all about the type of costs, and the various discount programs that cloud providers offer, and we built a cost model. The latter is an extensive exercise, where we list all components that we use in our clouds.

In multi-cloud, an application could span services over various clouds. By choosing the perspective of the application, all services will be captured. If we have done cost modeling right, we will know exactly what resources are connected to the application and, with that, what running the applications costs. Then there’s one last thing to do and that’s coupling the application to the application owner, and with that, the team, department, or business unit.

Before we can start the process of chargeback, we need visibility on the application chains, the resources and services that are used in the cloud, and the respective costs that are associated with these resources and services. This process is referred to as **showback** in FinOps: analysis of all data about usage in the cloud, enabling reporting to relevant stakeholders. This data must cover all aspects that we discussed in the previous sections: usage, cost elements, rates, and discounts applied to rates. This data must be identifiable to an application or an application chain and, with that, to an application owner, team, or business unit.

The following step is **chargeback**, where the costs are sent to the accountable part of the organization and charged to the budgets of a team or a unit. The challenges here are the shared costs and the way discounts are distributed over the various resources, including the shared resources. This will require financial engineering, one of the key tasks of the FinOps team that we set up in the previous chapter. The first step is to be able to allocate the costs to the right budget. In a truly mature organization, the chargeback is automated and integrated into the financial system, and shared costs are allocated based on the actual usage of the shared resources.

This concludes the part about financial operations in the cloud. The next part of this book will be about security and, specifically, implementing security using the DevSecOps principles.

**Summary**

In this chapter, we learned how to build a cost model that allows us to track costs in the cloud and how these costs can be allocated to specific budgets. The key is the understanding that we must cover all resources and services that an organization uses in the cloud, including all shared resources.

We also saw that to get the right costing level, we must consider that we’re dealing with various types of rates. The rates that are shown in the pricing calculators of cloud providers are often not applicable: discount programs might be applied, influencing the true costs of cloud environments. We learned about blended and unblended rates, managed and unmanaged services, direct and shared costs, and consumption-based and fixed costs. All of this must be reflected in the cost model to get to an accurate showback and eventually chargeback.

This was the final chapter about FinOps. The next part of this book will be about security.

**Questions**

1. What is the main benefit of reserved instances for companies?
2. On invoices, various rates for the same type of resources might appear. What do we call these types of rates?
3. Microsoft offers a program that allows for the optimal use of licenses for Windows Server, SQL Server, and various Linux distributions in Azure. What is this program called?

**Further reading**

* The FinOps Foundation releases the *State of FinOps* yearly, showing the trends in cloud financials. The *State of FinOps* can be found at <https://data.finops.org/>.