**11**

**Maturing FinOps**

In the previous chapter, we discussed the principles of cloud financial management, or FinOps. Now we know how we can provision workloads and track the costs of these workloads, it’s time to take the next step and professionalize our FinOps practice. We will learn how to improve cloud financial management by studying various maturity models.

This chapter is about the transformation to managed FinOps in our organization, by setting up a FinOps team that has a major task in the adoption of the FinOps principles. Adoption starts with awareness. Hence, we will learn how to integrate FinOps principles in a cost-aware design process, making developers and engineers aware of the fact that every decision they take and every resource they implement has a financial consequence. That’s OK, as long as the implemented solutions add to business value. That’s the key theme of FinOps.

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

* Setting up a FinOps team
* Using maturity models for FinOps
* Introducing a cost-aware design
* Transformation to managed FinOps in multi-cloud
* Avoiding pitfalls in FinOps transformation

**Setting up a FinOps team**

For starters, FinOps is not about saving money in the cloud, but about making money with the cloud. Hence, it’s tightly integrated into business planning and the forthcoming architecture. FinOps has proven to be a must to stay in control with the entrance of cloud and DevOps.

The challenge is evident: where engineers used to order physical equipment and had to go through a financial approval process to get that equipment delivered to a data center, now they can get that virtual equipment with a click of a button. However, although the equipment is virtual and basically just a piece of code deploying a resource in the cloud, it still costs money. The main difference is that companies shift from CAPEX to OPEX.

**CAPEX**—**capital expenditure**—concerns upfront investments; for example, in buying physical machines or software licenses. These are often one-off investments, of which the value is depreciated over an economic life cycle. **OPEX**—**operational expenditure**—is all about costs related to day-to-day operations and, for that reason, is much more granular, meaning that costs can be more detailed and defined per resource. Usually, OPEX is divided into smaller budgets that teams need to have to perform their daily tasks. In most cloud deployments, the client only pays for what it’s using. If resources sit idle, they can be shut down and costs will stop. A single developer could—if mandated for this—decide to spin up an extra resource if required.

That’s true for a **pay-as-you-go** (**PAYG**) deployment, but we will discover that a lot of enterprises have environments for which it’s not feasible to run in full PAYG. You simply don’t shut down instances of large, critical **Enterprise Resource Planning** (**ERP**) systems such as SAP, just as an example. So, for these systems, businesses will probably use more stateful resources, such as reserved instances that are fixed for a longer period. For cloud providers, this means a steady source of income for a longer time and therefore, they offer reserved instances against lower tariffs or apply discounts. The downside is that companies can be obliged to pay for these reserved resources upfront. Indeed, that’s CAPEX. To cut a long story short: the cloud is not OPEX by default. The best advice here is to check the conditions with the cloud provider.

Hence, the journey to the cloud will require financial expertise. At least, that is strongly advised for businesses. It’s not a matter of just pulling out a credit card to get started: businesses need a plan, just like they did when they still had data centers. Cloud is about doing investments with the right business justifications. That’s the reason why every **Cloud Adoption Framework** (**CAF**) has governance and finance as one of the essential pillars.

We need the cloud, financial experts, and a plan. The FinOps Foundation calls this “laying the groundwork.” This includes the collaboration between engineers, finance, and executive leadership. Technical and financial experts need to work together in moving environments to the cloud. But someone has to take the lead: this role is called the **Driver**.

In this book, we will use the terminology of the FinOps Foundation as the industry standard. More information can be found on the official website of the foundation: [finops.org](http://finops.org/).

The Driver is the persona that glues the required disciplines together and drives the process of establishing the FinOps practice. This role is fulfilled by a FinOps practitioner. The main task is to advocate the principles of FinOps and develop the capabilities. This is not something that the FinOps practitioner can do in isolation. A good FinOps practice takes a FinOps team that works closely together with product or application teams, IT, finance, and the business.

The main tasks of the FinOps team are:

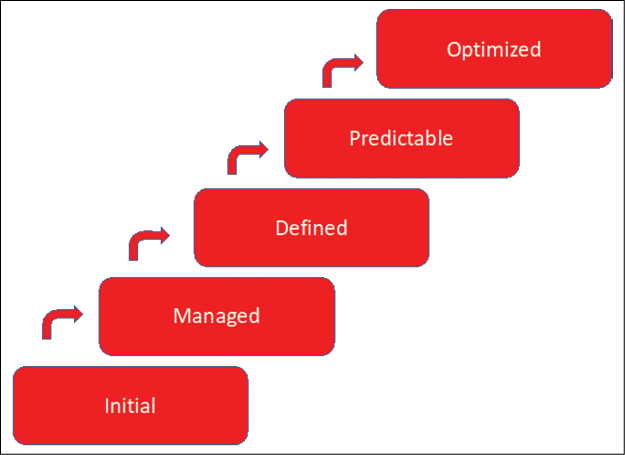
* Setting up control guidelines for cloud costs.
* Be aware that in multi-cloud environments, there can be different guidelines for different clouds. Hence, the team must have knowledge of the clouds that are used by the business. Refer to *Chapter 10*, *Managing Costs with FinOps*for more detail.
* Setting standards for tagging.
* Setting standards for cloud cost allocation.
* Businesses will have divisions, units, and teams who all can provision resources in the cloud and initiate costs. These costs must be allocated to the right team: the team that is responsible for the resources. This is where chargeback models are used. This is the topic for *Chapter 12*, *Cost and Value Modeling in Multi-Cloud Propositions*.
* Matching budgets with actual spends.
* Collaborating with application and development teams to improve efficiency in cloud resource provisioning, including purchasing strategy, and right-sizing of resources and automation.

The FinOps team must be knowledgeable about the cloud and cloud technologies, but also about buying and licensing strategies of the cloud providers. This team is typically the team that views the costs, validates the invoices, and advises application teams and businesses on improvements, leading to cost efficiency and more added business value. This is not an easy task.

A good FinOps practice is not set up in one day. Businesses will need to grow the FinOps practice. Maturity models will help with this. We will study the models in the next section.

**Using maturity models for FinOps**

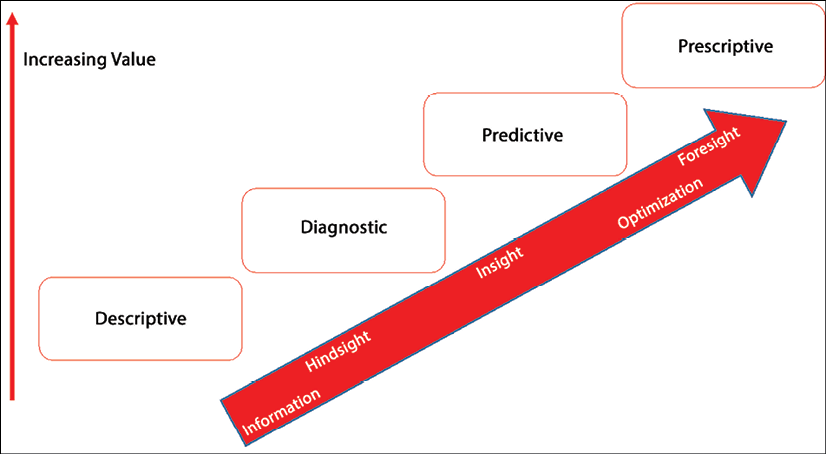
In *Chapter 3*,*Starting the Multi-Cloud Journey*, we discussed the generic maturity model as guidelines for an assessment, defining how mature an organization is and what its ambition is. We used the **Capability Maturity Model** (**CMM**) as an example. It’s shown once more in the following diagram:



*Figure 11.1: The Capability Maturity Model (CMM)*

We can plot this model on cloud financial operations too. The principles are the same, where on the initial level processes are poorly controlled, and outcomes—in financial terms, this would refer to costs—are highly unpredictable. Level 5 is the highest level, where organizations can focus on improvements since projects and management are well-defined, processes are under control, and costs are predictable and measurable.

Let’s look at another maturity model that will help in establishing financial management. This model is inspired by an originally developed model by the research company Gartner.



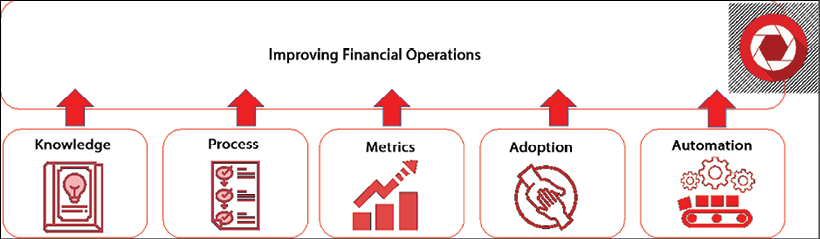
*Figure 11.2: Maturity model showing stages from information to optimization*

The main themes are hindsight, insight, and foresight, and that’s exactly what FinOps aims to do: provide insight and enable foresight through financial analysis. It means that we need data and processes to analyze the data. We need metrics to validate the outcomes of the data and eventually predict future cost developments. Using automation, we could build in processes to automatically perform actions in our clouds to stay in control of costs.

These steps—hindsight, insight, and foresight—are integrated into the maturity model that was developed by the FinOps Foundation. That model has five lenses:

* Knowledge
* Process
* Metrics
* Adoption
* Automation

They are shown in the figure:



*Figure 11.3: Maturity—The five lenses for improving FinOps*

Knowledge, process, and metrics are basically about the vocabulary in financial operations. It’s about having one common language that architects, engineers, financial experts, and management understand. If all stakeholders speak the same language, have the same information and data, and know how to interpret that information and data, combined with shared goals, then adoption will be easier.

Adoption requires a common understanding of data, a commonly shared vision of how business should evolve, and how IT can help enable this vision. The following step would then be to optimize; in the case of FinOps, this should be through automation of processes, for instance, using mechanisms in the cloud that automatically scale resources to align them with budgets and forecasts in usage of resources.

But how do we get there? The FinOps Foundation developed a methodology with three steps or stages: crawl, walk, and run. This is a very simple, yet comprehensive way to grow the maturity in implementing and applying FinOps. The best part of it is probably that it is completely agnostic to any cloud platform. The model simply tells a business how to grow in complexity and scope of the cloud services and manage the costs that are associated with these services, on any platform. An organization will typically go through three stages:

* **Crawl**: In this phase, the organization starts applying the basics of FinOps. There’s very little reporting and only on basic metrics. Measurements can be made, and costs can partly be allocated to the business, by mapping resources to business processes or products. Reporting is only in hindsight, and forecasting versus actual cloud spend is not accurate, meaning that there are significant deviations in actual spend and forecasting.
* **Walk**: A much larger part of the cloud spend can be allocated to business functions such as sales and delivery processes of products. Collecting data for metrics is already largely automated.
* **Run**: Cost allocation is almost entirely automated, where cloud resources can be allocated to business functions and cloud spend is automatically charged back to that business function. Accuracy in forecasting versus actual cloud spend is high.

The big question now is: how do we take this into real-life practice? In the next section, we will try to provide some guidelines.

**Introducing cost-aware design**

This book, and even this specific chapter, started with the observation that implementing cloud solutions must add value to the business. It’s the most important principle to keep in mind when we’re discussing financial operations. FinOps is not about cutting costs, but about adding value using cloud technology. But inevitably, we will have to deal with costs. The cloud simply doesn’t come for free.

As such, we need to know what the costs are when we design for the cloud. Next, we need to identify risks, since these will also inflict costs when risks materialize. These can be direct costs, but also costs that help us mitigate the risks. Lastly, we must define the expected benefits when we design and implement a solution. Costs, risks, and benefits will facilitate the calculation of the **Total Cost of Ownership** (**TCO**) and the **Return on Investment** (**ROI**), as shown in *Figure 11.3*.



*Figure 11.4: Enterprise TCO/ROI model*

The benefits are all about the business outcome or the business value. The business value can, for instance, be increased revenue or faster time to market. If we look at the Cloud Adoption Frameworks of the cloud providers, we will notice that all frameworks work toward the business outcome as the justification to go to the cloud. But these frameworks also address the risks and the costs that a company will make to move resources to the cloud.

What are the costs? First, we have direct costs, sometimes referred to as out-of-pocket:

* Licenses
* Costs for resources such as virtual machines, firewalls, gateways, and other infrastructure services in the cloud
* Time that is spent by architects, engineers, and other professionals on designing, implementing, and operating cloud environments

There might also be hidden costs or costs that can’t directly be associated with resources in the cloud. Think of the training of staff.

In terms of risks, we have to mention one particular tricky element: software licenses that we use in cloud environments.

When an enterprise uses software, it needs to purchase licenses for its usage. That goes for both proprietary and open-source software. The main difference between these two categories is that open-source software does allow modifications and changes in the software, as long as the changes are committed back to the development of that software. Proprietary software is typically closed source, where the source code may not be modified.

As with a lot of licenses, this can become quite complicated. If a product that a company uses is not sufficiently licensed, the company can be forced to pay fines along with the license fees it should have paid from the start. Using non-licensed software is illegal. That doesn’t change when we’re moving environments to the cloud.

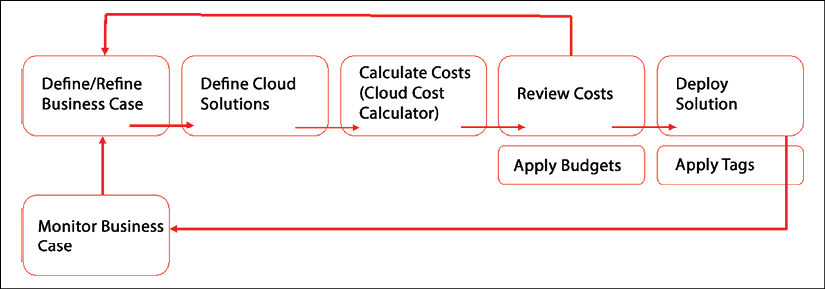
How do you know when a software product is properly licensed? There are just a few types of software licenses—with a lot of variations—but stripped to their essence, they come down to these categories:

* **License on a user basis**: This is often the model that is used for end user licenses and it’s probably the most straightforward way of licensing. For each user, there’s a license that entitles the use of the software. A good example of this is Microsoft 365, for which a company can order a license per user, per month. In that case, there’s a one-to-one relationship between the user and the use of the specific software product.
* **Licenses based on resources**: This is a more complicated licensing model. An example is software licenses based on the usage of a specific number of CPUs or the number of database instances. This is still a popular way of licensing proprietary software: the license fee is calculated according to how many CPUs or instances are being used. The issue in cloud infrastructure is that resources are quite often shared resources, virtualized on top of the real physical machine. Which CPU is licensed then? Typically, the license in cloud environments is based on the virtual CPU, in that case. Keep in mind that support agreements on software might change as soon as the software is deployed in cloud environments. Not all software is supported when used in Azure, AWS, or Google Cloud Platform.
* **Lump sum fee**: Software is purchased and paid in full upfront. This may even apply to major updates, whereas in subscription-based models, you will have access to major updates too. For software vendors, the lump sum is not a very attractive model, since they will only receive a one-time payment. But for companies, this is also not very attractive, since they will be confronted with high, upfront, cash-out investments.

How do you know when a software product is properly licensed? For that, specialized **Software Asset Management** (**SAM**) tools can be very convenient. SAM tools do a lot more than just make an inventory of all the software that a company uses: these solutions evaluate the whole life cycle of software, from purchase to deployment and, indeed, to utilization.

Other risks are an increase in costs or a lack of skilled staff. Both are becoming more and more a fact of life than a risk. A lot of companies already suffer from a lack of skilled personnel and are confronted with increasing costs. It’s obvious that risks will influence the business value severely when they materialize.

To manage costs and risks we need a managed process. That process starts with a **cost-aware design**. The process is represented in the diagram.



*Figure 11.5: Cost-aware design process*

The process begins with the business case and then helps the business to stay in control of costs and investments in the cloud by executing the following activities. Note that all activities loop back to the business case:

* **Mapping the data about cloud spend to the business**: An organization will have cost centers, business units, application owners, and, of course, applications and data. This is represented in an organizational breakdown. The cloud spend must be mapped to this organizational hierarchy.

This will take a lot of time, but it must be done. This is typically an activity to start with in the crawl phase, but it can become more granular and detailed in the next phases. It’s essential that every dollar that is spent in the cloud is accurately allocated to a cost center, unit, or application owner. First, it’s important to agree on how the organizational hierarchy is reflected in the design of the cloud platforms.

* In multi-cloud, we have to ensure that this is done identically in each of the platforms that we use. The process of setting up account hierarchy must be the same across the platforms, otherwise, the recharging model won’t work. Costs will not be accurately allocated or, worse, completely missed in the models and left unallocated.
* **Define tagging strategy**: This must be aligned and applied throughout the entire organization. Agree on what type of tags are mandatory in every cloud platform. Make sure that tags allow for accurate reporting, meaning that resources are uniquely identifiable through tags and show up as unique resources in financial reporting. But tags should also allow fast, fault-tolerant allocation of the specific resource.
* **Define a clear showback and chargeback model**: When we have an organizational structure in place and agreement on the tagging policies, we can define a showback and chargeback model. Showback will clearly show what costs are made by a specific application, unit, or cost center; in other words, who is accountable for cloud spend. Chargeback will actually push the costs to that organizational unit, charging these costs to the budgets of the unit. In the next chapter, we will discuss this in more detail.
* **Define budgets and forecasts**: In a later stage, organizations will be able to use data about cloud spend to do accurate forecasting and set budgets for the organizational units that we defined in the first step of our cost-aware design. This can be automated too, for instance, by automated sizing of resources or suspending resources when these are not used for a certain time. By setting budgets, we can get alerts when resources in the cloud reach thresholds. All this information can next be used again for adjusting forecasts.
* Forecasts must be done for every organizational unit and must take every aspect or artifact in the cloud into consideration. Hence not only the use of virtual machines but also containers and serverless functions. Containers and functions are workloads too and cost money.

Now we can start building a cost model. This will be discussed in detail in *Chapter 12*, *Cost and Value Modeling in Multi-Cloud Propositions*. First, we will learn how to get operational excellence by implementing FinOps.

**Transformation to managed FinOps in multi-cloud**

In terms of maturity, we have reached the level of understanding financial management in the cloud and how we can set up a cost-aware design process. But our challenge is multi-cloud. We will likely have to deal with more than one cloud platform, which makes cloud financial management even more complicated. Still, we can use the FinOps principles to get to managed multi-cloud financial operations.

The FinOps Foundation provides a comprehensive model for managed operations, using three stages: inform, optimize, and operate. We will recognize this from the model that we showed in *Figure 11.2*.

This is a continuous cycle and part of the entire cloud operations. In multi-cloud, this implies that this cycle must be fulfilled for every cloud platform that we are utilizing. In multi-cloud, we need a single pane of glass, covering all the platforms in our landscape.

First, let’s explore the three phases in the cycle:

* **Inform**: This phase is about visibility and getting the right data. This is actually harder than it sounds in the cloud. Cloud is by default dynamic. We have discussed this in the first two chapters of this book: companies migrate to the cloud as part of their digital transformation, making the business more agile. The development of new products can be accelerated because companies can respond faster to new market demands.
  + Companies can lift and shift their traditional IT into the cloud without making use of the real advantages of cloud and cloud-native; in that case, the environment in cloud platforms will remain rather static and data about the environment will not change much. But if cloud-native services are used with, for instance, automatic scaling, then the environment will be very dynamic. This has to be constantly verified against the business case: does the cloud deliver the expected business value that we defined during our cost-aware design? Continuously gathering the right data is crucial to enable optimization in the cloud.
* **Optimize**: This is the phase where companies can optimize their cloud environments. This is not only about enhancing automation, but also about planning and forecasting. That’s why the inform phase is so important. If companies use specific workloads for a longer period, it might be worthwhile to consider a different deployment schedule for these workloads. Cloud providers offer reserved instances or committed use discounts (as Google refers to reserved instances) that have financial benefits in cloud consumption. A workload on a reserved instance in the cloud can save up to 70 percent in costs. But it means that the provider can’t use the reserved compute and storage capacity for other companies anymore, since it’s now reserved and committed.
  + The provider will likely ask for a sort of guarantee that the customer will consume the resource for one, three, or five years. The provider might even require upfront investments to cover this decrease in capacity that now can’t be freely allocated. Of course, providers will have mechanisms in place to still enable shifting with capacity and reserve part of the capacity for one customer as reserved or committed use discount. There’s also an advantage for the providers since they are assured that a customer is committed to using the cloud for a guaranteed period.
  + Optimization can also be resizing and even redesigning workloads. Resizing can be done automatically. If a workload is hosted on an instance that is only using half of the compute and storage capacity in normal operation, automated scripts or the autoscaling capabilities provided by the cloud can reduce the compute and storage capacity. When usage peaks, the capacity can be scaled out again, also automatically.
  + The redesign will require an evaluation of the architecture. Using cloud-native services, such as serverless or migrating applications to containers instead of full-size VMs, can lead to a decrease in costs and, more importantly, lead to more flexibility, increased efficiency, and better performance. As an end result, this can lead to better business results—the main goal of FinOps. However, this is often not an easy task.
* **Operate**: The data that we gather in the inform phase, leading to optimization, must be continuously evaluated against the business case and with that, the business goals. In the end, it’s all about knowing what’s going on, delivering the right resources to achieve business goals, and continuous improvement by constant optimization, and by doing all of this, increasing value to the customers. In a nutshell, this is operational excellence and FinOps can have a great attribution to achieving that excellence. FinOps can help in monitoring metrics that indicate how business objectives are met. It is important to have clearly defined metrics, formulated in a **SMART** way: specific, measurable, achievable, relevant, and timely. The FinOps team that we discussed in the first paragraph of this chapter can make this happen, working together with all other stakeholders, including the business.

The next step is to implement that single pane of glass since we are dealing with multi-cloud. We need tools for that. VMware’s CloudHealth and Apptio’s Cloudability are the leading tools in the space of financial management in multi-cloud. They offer real multi-cloud visibility and with that, the starting point for optimization and business-case-driven operational excellence:

* **CloudHealth by VMware**: **CloudHealth** will help organizations in managing clouds across the leading platforms AWS, Azure, GCP, Oracle, and, of course, **VMWare**. It monitors cloud spending and provides advice in optimizing the cloud resources with right-sizing and the purchase of reserved instances. It follows the FinOps maturity model with visibility, optimization, and finally, business integration in operations. More information about CloudHealth can be found at <https://cloudhealth.vmware.com/>. Be aware that the branding recently has been changed to VMWare Aria.
* **Apptio Cloudability**: **Cloudability** by **Apptio** works cross-platform too and provides extensive dashboards on cloud spend, budgets, and forecasting. It extracts billing and consumption data from the cloud platforms, while allowing business mapping. This can be very finely granular, up to the team level, which makes the tool suitable for use in DevOps streams. A strong asset in this tool is anomaly detection with alerts when resources overrun budgets and forecasting without direct reasons. More information on Apptio can be found at <https://www.apptio.com/solutions/cloud/cost-management/>.

We can have all the best-in-class tools and have our organization set up according to the best practices of FinOps, but there are still a few pitfalls that we should avoid. This is the topic of the next section.

**Avoiding pitfalls in FinOps transformation**

Let’s start by repeating this message: FinOps is not about cutting costs in the cloud. It’s about adding value to the business through cost management in the cloud. That is a major difference. The mindset of cost-cutting can lead to unwanted situations that can cost a company a lot of money. Think of this example: development environments that are not used during the night might be switched off during the nighttime and brought back online in the morning when developers get to work again. This will save costs, but the question is if it brings value to the company when developers can’t work on environments outside of office hours.

Note: This is not advocating that developers should work 24/7, but there might be occasions where it’s OK to allow people to work at times that are convenient for them. It raises productivity.

In international companies, shutting down instances in a specific timeframe might even be completely impossible. Think of European or American companies that have their development outsourced to India or other offshore countries.

One other major pitfall is buying a tool for cloud management without having the processes and people in place. There are first steps that must be taken: getting the right people on board, setting the objectives, and defining the metrics.

The most important lesson in implementing FinOps is probably that it should not be done following a top-down approach. It will not lead to the adoption of FinOps practices, and we won’t reach our objectives: making teams that work in cloud platforms aware of financial consequences. Teams should have a great responsibility in using the right resources and services in the cloud, but also must be cost-aware. Every design decision that they take has a financial consequence. That is awareness and it’s the start of adoption.

We have learned how to grow the maturity in FinOps. In the next chapter, we will start building cost models in the cloud, using all the different deployment and service models in multi-cloud environments.

**Summary**

In this chapter, we discussed how we can professionalize our FinOps practice by using maturity models. The key learning was that teams that work in the cloud must become cost-aware. We studied the principles of cost-aware design and how the maturity lenses of the FinOps Foundation can help us in adopting this way of working and, eventually, achieve operational excellence in managed FinOps.

We learned that getting the right data on resources and cloud spend is crucial and that we should map this to business processes and functions using showback and chargeback. From there, we can start optimizing our cloud resources, for instance, through right-sizing and purchasing reserved capacity. We have stressed that all these activities must lead to increased business value, rather than cutting costs in the cloud. In the final paragraph, we discussed a few major pitfalls.

We are now ready to go one step deeper in financial management in multi-cloud, and that’s developing cost models. This will be the topic of the next chapter.

**Questions**

1. Name the five lenses in the FinOps Maturity Model.
2. What are reserved or committed use discount instances?
3. True or false? FinOps is mainly about cutting costs in cloud usage.

**Further reading**

* Information about and certification for FinOps practitioners can be found on the website of the FinOps Foundation: [finops.org](http://finops.org/).