

Shifting FinOps Left

Empowering Engineers to Take Action



Table of Contents

Foreword.....	3
What is FinOps.....	4
Your FinOps Roadmap.....	10
Evolving Cloud Dynamics.....	16
Shifting FinOps Into Engineering Workflows.....	23
A Real-World Case Study.....	36
Better Outcomes For Companies And Engineers.....	41

Foreword



Hassan Khajeh-Hosseini

Co-Founder & CEO

Infracost



Ali Khajeh-Hosseini

Co-Founder & Head of Product

Infracost



Alistair Scott

Co-Founder & Head of Engineering

Infracost

In 2011, Ali, Alistair, and I started working on a startup that could help companies calculate the Total Cost of Ownership (TCO) of adopting Amazon Web Services. That startup was soon acquired and evolved into one of the first cloud cost management products in the market (and is now part of the Flexera cloud cost management product).

Since 2011, we have seen quite a few changes in this market, some big and some small. However, we have still been unsatisfied by the lack of focus on helping engineers make better decisions. More often than not, engineers are the infrastructure buyers, and you must empower them with the information they need at the right time, place, and granularity to make better decisions. Infracost exists to empower engineering teams to use cloud infrastructure economically and efficiently.

We launched Infracost in 2021. In less than three years, Infracost has been adopted by over 3,000 companies, helping organizations worldwide to shift FinOps left. Today, Infracost is used in some of the world's largest cloud deployments.

We wrote this e-book to help you kickstart FinOps procedures in your organization. Additionally, we will show you how some of our customers use Infracost specifically to save their engineering teams time and prevent unnecessary cloud costs and waste from going into production in the first place.

What is FinOps

FinOps is a methodology that brings financial accountability to the flexible spending model of cloud computing. However, this description only scratches the surface of what FinOps entails. The shift towards cloud computing brings about a cultural change that moves decision-making power out to the edges of an organization, from procurement to engineering, architecture, and product teams. FinOps empowers technology, finance, and business professionals to collaborate more efficiently to maximize the business value from using the cloud.

Businesses worldwide are rapidly transitioning to cloud computing, regardless of their preparedness. This shift, akin to previous technology changes like the move from mainframes to client/server, web, or mobile, entails significant modifications to conventional business practices for accounting, requesting, and managing technology costs. As a result, there is a transformation from project-based to product-based operations.

FinOps wants you to understand that the traditional ways of managing infrastructure are not just ineffective, but they can also create situations where uncontrollable cloud expenses can put your business at risk. While traditional infrastructure management methodologies will still be necessary for owned infrastructure in the future, following these methodologies alone won't help you manage cloud expenses efficiently as they are not well-equipped to handle the variable consumption model of the cloud. Therefore, to manage cloud expenses effectively, you need to introduce FinOps.

The FinOps Foundation has defined "FinOps" as "an operational framework and cultural practice which maximizes the business value of the cloud, enables timely data-driven decision making, and creates financial accountability through collaboration between engineering, finance, and business teams." (this is the updated definition as of December 2023, and may change again in the future).

What is FinOps Framework

FinOps Framework

FinOps is an operational framework and cultural practice which maximizes the business value of cloud, enables timely data-driven decision making, and creates financial accountability through collaboration between engineering, finance, and business teams.



Principles

- Teams need to collaborate
- Decisions are driven by business value of cloud
- Everyone takes ownership for their cloud usage
- FinOps data should be accessible and timely
- A centralized team drives FinOps
- Take advantage of the variable cost model of the cloud

Domains & Capabilities



FinOps.org

Core Personas



Allied Personas



Phases



Maturity



Why FinOps Matters

Most organizations grapple with escalating cloud costs as they rapidly scale their operations. Once touted for cost-efficiency, the cloud is a significant line item in the budget. A new operational model called Cloud FinOps has emerged to tackle this issue. This model involves a combination of technology, business, and finance professionals working together to manage and optimize cloud spending.

At its core, FinOps is about extracting maximum business value from every cloud dollar spent. It blends the expertise of financial processes with the agility and responsiveness that cloud services offer and involves all stakeholders to be held accountable.

THREE MAIN BENEFITS

TRANSPARENCY

FinOps allows clear visibility into cloud spend and usage for all parts of the organization, from engineers to engineering management, finance, procurement, and executive teams.

BUSINESS ALIGNMENT

It aligns cloud spending with business strategies. This includes training teams, calculating and bringing unit economics to the organization, defining and increasing maturity, and applying the principles.

OPTIMIZED COSTS

Achieve effective cloud cost optimization by continuously monitoring and adjusting your cloud operations. It comes down to saving money and increasing cloud utilization.

Cloud Cost Optimization: A Formula

Cloud Cost 💰 = **Usage** 🛠️ × **Unit Price** 🏷️

↳ The bill

↳ Engineering

↳ Finance / FinOps

↳ Decentralized

↳ Centralized

Understanding the formula for cloud costs is the first step in creating an effective cost optimization strategy. Essentially, cloud costs result from the multiplication of usage and unit price. Simply put, how much you use (cloud usage) multiplied by the cost for each unit of that service (unit price) constitutes your total cloud cost.

Hence, effectively managing either part of this equation or both can significantly influence your overall cloud expenses. You can optimize your cloud costs effectively by minimizing excessive usage and choosing cost-efficient unit prices.

The Need to Shift FinOps Left

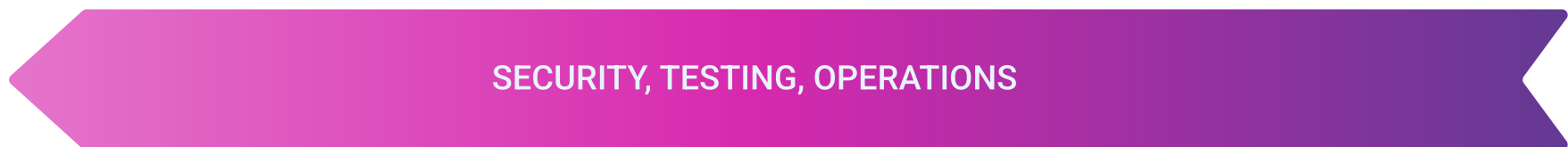
Introducing processes earlier in the software development cycle, known as "shift left," has become popular in software engineering and DevOps.

The shift-left principle originated in software testing. In a traditional waterfall model, testing is performed just before the release. However, in shift-left testing, testing starts earlier by introducing practices such as Test-Driven Development (TDD) and Behaviour-Driven Development (BDD).

The shift-left principle is now applied to various other disciplines. The term "DevSecOps" has been coined to introduce security early in the software development cycle, creating a whole new ecosystem of tools to help implement this practice.



Shift Left



By shifting FinOps left and putting costs directly in engineering workflows before resources are launched, we can achieve proactive FinOps.

There are **two** main benefits of doing this:

1. Cost Avoidance

Once we shift cloud costs left, then engineers can catch costly mistakes before they ship their code. This means the overspend never goes into the cloud bill in the first place. This is cloud cost avoidance.

2. Engineering Time Saved

The earlier we catch and fix issues, the more efficient the engineering processes are. A good example is spotting issues with tags. If we have resources that are not tagged properly, and we catch these issues in the engineering workflow, the fix is measured in seconds. However, if the tagging issue goes to production, and is discovered later, the fix time is measured in hours.

Your FinOps Roadmap

1

In the previous chapter, we discussed why FinOps matters. Now, let's talk about how to establish a foundation for FinOps procedures in your organization.

Step 1

Assemble Your FinOps Team

Why Cross-Functional Teams are Crucial

The FinOps framework heavily advocates for cross-functional teams. This assembly involves representatives from engineering teams, financial teams, and business operations. Having multiple departments represented ensures that all perspectives on cloud costs and usage are considered, leading to a more holistic cloud financial management strategy.

Management loves to see results, so start with a small, super-capable team to validate and prove that FinOps is an excellent addition to managing and optimizing your organization's cloud spending. The team doesn't have to work full-time on this initiative initially; that happens much later.

Roles Within an Initial FinOps Team

Of all the personas of the FinOps foundation, "Engineering" and "Finance" are the key members needed in the first phase. Later, you will build bigger, cross-functional teams that will collaborate.

Engineering: They will understand the current and future state of the infrastructure and manage cloud workloads. The engineering teams can help navigate which resources are easy to optimize and which will require more work. This will be the most critical role in the initial FinOps team.

Finance: Beyond budgeting and forecasting, the finance team tracks spending models, conducts cost-benefit analysis, and negotiates with cloud providers.

Your FinOps Roadmap

2

Your main goal is to first target the largest expenses on your cloud bill. Identifying cost-saving opportunities is crucial to pitch wider FinOps adoption.

Step 2

Conduct an Initial Cloud Spend Assessment

How to Analyze Cloud Expenditure

The cornerstone of effective cloud financial management lies in an exhaustive analysis of where the dollars are going. This step involves auditing your cloud resources, services, and spending patterns. Understanding your cloud spending will highlight areas for cost savings and potential optimization.

Tools for Addressing Cloud Costs

Cloud providers such as AWS, Microsoft Azure, and Google Cloud Platform offer native tools for tracking expenses. It is best to start there. These free tools provide a good starting point to identify the top cloud cost-saving opportunities.

Your FinOps Roadmap



3

Step 3

Implement Standard FinOps Practices

Following FinOps Principles

FinOps principles serve as the compass for implementing best practices. These principles include proactive monitoring of cloud resources, introducing accountability for cloud usage, and adopting a culture of continuous optimization.

Toolkit for FinOps Implementation

Many tools can help you on your FinOps journey. These range from open source software, which you must deploy and maintain, to commercial paid products to help analyze your bills.

The key to these tools is to use them to help your engineers make better decisions. It's always better to prevent cloud costs from going onto your bill in the first place rather than relying on bills to identify fixes. This is why we've built Infracost – we make FinOps proactive by sitting directly in the engineering workflow, and every time an engineer makes a code change, we tell them the cost impact of the change and scan their code to ensure FinOps policies are followed, and all resources are tagged correctly.

Your FinOps Roadmap

4

Step 4 Establish Robust Financial Processes

Implementing Cost Allocation Techniques

Cost allocation is not just an accounting exercise. It's a critical component of cloud financial management that drives accountability across various business units and departments. By correctly allocating costs, you can pinpoint inefficiencies and understand how different areas of your business contribute to cloud spend. The key to making cost allocation work is to have a good proactive tagging policy.

Advanced Cost Optimization Strategies

Beyond the basic tactics like reserved instances or shutting down idle resources, consider employing complex strategies like heat mapping usage patterns to allocate resources more efficiently. There are many other advanced optimization strategies, but these will require architects, engineers, and finance to work together.

Your FinOps Roadmap

5

Step 5 Continuous Measurement and Optimization

Key Performance Indicators (KPIs)

In a well-run FinOps practice, regularly tracking KPIs such as cost savings, cloud waste reduction, and business value generated per dollar spent on cloud services is critical for ongoing success.

The Cycle of Reporting and Feedback

Feedback loops should be established between the FinOps team and business units to maintain constant communication. Routine reporting using dashboards accessible to all relevant stakeholders fosters a culture of continuous improvement.

If you put cloud costs into people's existing workflow (CI/CD, Jira, etc.), half the battle has already been won.

Your FinOps Roadmap

6

Step 6

Scale Your FinOps Practice

As your cloud deployments grow, your FinOps practices must scale accordingly. Whether dealing with multi-cloud environments or expanding your cloud workloads, having a scalable FinOps practice ensures that the variable cost model is maintained effectively.

Challenges in Scaling FinOps

Expanding your FinOps practice can be challenging, as it involves maintaining cost visibility at scale, ensuring robust cloud governance mechanisms, and managing the complex pricing structures that come with a larger cloud footprint.



Evolving Cloud Dynamics: Navigating Cost Management Challenges

As organizations delve deeper into their FinOps journey, it's crucial to understand the broader context of cloud adoption and cost management. The evolution from traditional infrastructures to dynamic cloud environments has revolutionized how we deploy technology and account for its costs.

In this section, we explore the transition from predictable, fixed costs in traditional IT to variable and sometimes unpredictable costs in cloud environments, highlighting the need for a strategic approach to cloud cost management.

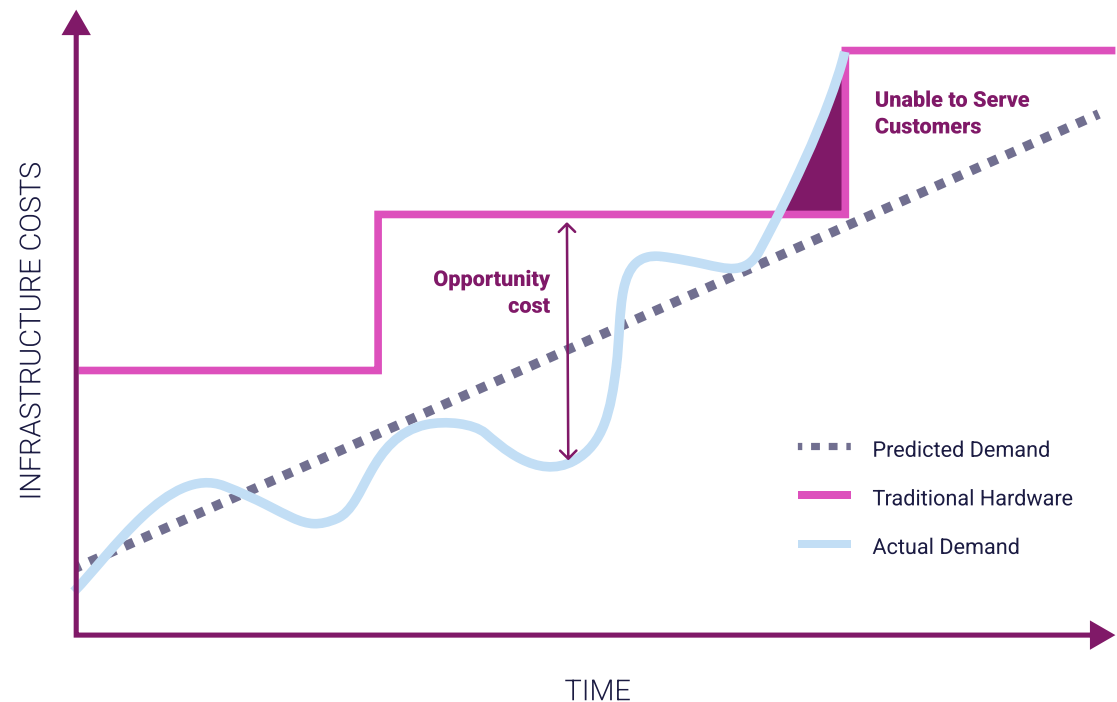
Cloud Enablement

Previously, we discussed that in a cloud operating model, the decision-making power regarding what infrastructure resources are used, when, and by whom is moved to the edges of the organization. Practically, any engineer can purchase and launch cloud infrastructure when needed.

One of the big benefits of using public clouds has traditionally been cost. The argument goes that in a traditional data center, you have to provision hardware to cater to the maximum peak of usage and "waste" those resources when they are not being used. In a public cloud environment, you only provision what you need (actual usage) and pay for what you use.

However, this only applies if you can match your usage with demand. What usually happens is that the usage and demand mismatch. Therefore, costs can also become a downside of using public cloud providers.

There are many benefits to using the public cloud: products are delivered faster to market as barriers are removed from the software engineering release process. Product teams are much more agile and flexible, which leads to innovation and quick iterations; productivity increases as engineers can use higher-level product dependencies that are maintained and optimized by cloud providers.



Cloud Costs: A Disadvantage?



Organizations started to see cloud cost as a downside of using the public cloud. There were three main pain points that companies faced:

- 1** Public cloud costs grew exponentially and were uncontrolled compared to the traditional data center approach.
- 2** Public cloud pricing has become increasingly complex, with thousands of price points and constantly changing fields.
- 3** Companies struggled to monitor cloud usage and costs due to untracked Shadow IT.

In response to the challenges faced by organizations regarding public cloud pricing, a new software category called Cloud Cost Management emerged in 2012. The primary objective of companies in this category was to help their customers analyze their cloud provider bills more comprehensively. With these service providers' help, customers could break down their bills by various factors such as timeframes, accounts, services, labels, and more.

The issue was managing costs associated with cloud services, and the finance team was willing to pay to tackle this problem. In the past, finance could budget for computing infrastructure purchases once or twice a year, but now it appeared that the actual bill would exceed the pre-defined budget every month. The term "cloud bill shock" was coined to describe this new challenge.

Bill Analysis is Too Late

These vendors provided a service to show business units or engineering teams how much they spent on cloud computing. The finance team would create a report template to display the end users' bills for the last month.

This approach's main problem is that the generated reports only show what is currently in production, running, and costing money. This is not an effective way to manage software engineering projects as it is not aligned with the development cycle. Additionally, this approach is reactive, which means you will have to continue paying for resources until a fix is found instead of being proactive and identifying issues or overspending before going over budget.



Agile Software Development Life Cycle

This is how the issue is currently manifesting itself:

1

Engineers work on sprint 1 tasks and develop features and products. They then launch it into production. Resources cost money, and engineers start the first phase of sprint 2.

2

While sprint 2 happens, the resources launched in sprint 1 cost money; however, no one has yet created reports or looked at what is going on regarding cloud costs.

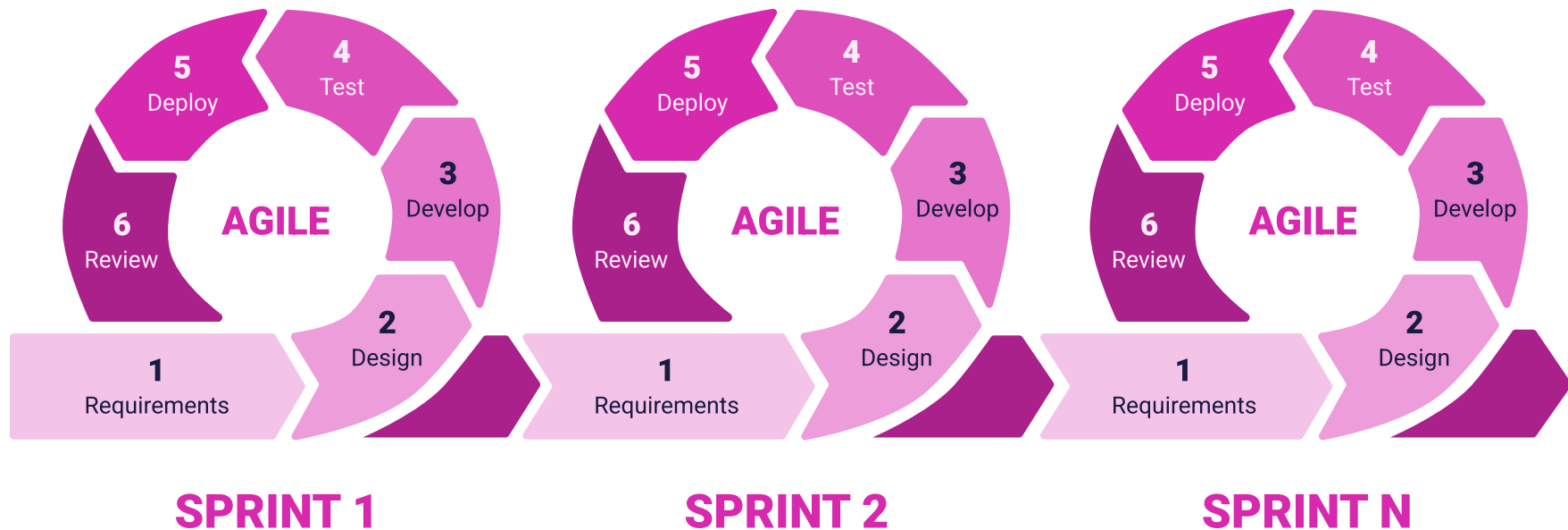
3

Sprint 2 finishes, and new resources are launched into production. Resources from sprint 1 are still running, and now resources from sprint 2 are also running and costing money. Engineering moves to sprint 3.

4

During the third sprint, an alert indicated that the cloud budget had been exceeded due to the resources launched in the first sprint. This raises a dilemma: should engineers stop working on the third sprint tasks to investigate the issues from the first sprint, or should they continue with the current sprint and allocate time in the next sprint to address issues from the first? However, neither of these options addresses the issues from the second sprint, which have yet to be identified.

Agile Software Development Life Cycle



This happens repeatedly. Engineers are frustrated because their work is constantly interrupted by issues from weeks or even months ago. Finance is frustrated because budgets keep breaking, and now they have to answer the why and try to re-do the budget.

An Analogy: The Supermarket

Imagine walking into a supermarket and being able to put anything you want into your shopping cart. But there's a catch: no price tags anywhere in sight. You fill up your cart and head to the checkout, only to find that there isn't one. Instead, you give them your credit card and are told you'll be charged for everything you've bought.

Later that month, you receive a hefty bill you do not know how to decipher. You have to go through all your shopping trips, find the bill, and analyze the data to see which products you bought and how much they cost. It's a frustrating process and a system we've accepted in our cloud computing world for over a decade.


It's easy to see that this is a broken system. It's time to change it!



The Key to Success: Shifting FinOps Into Engineering Workflows

The Core Issue: Reacting to changes seen on the cloud bill is too late

As discussed in the prior sections, we have missed one of the biggest opportunities when something is already in production and incurring costs. The issue here is that money has already been wasted, and fixing these issues is going to require investigation time, engineering time, and a whole release process to fix production. Shifting FinOps left is preventative care by nature, saving engineering time and preventing waste.



Shifting FinOps left is preventative care by nature, saving engineering time and preventing waste

What Changed? Why Can We Only Now Shift Cloud Costs Left?

While shifting left is not new, when it comes to FinOps, it was only possible to do it in a realistic matter once two fields advanced and became widely adopted. These are Infrastructure as Code (IaC) and DevOps.

Infrastructure as Code (IaC): IaC allows developers to specify and manage infrastructure using code, facilitating automation and consistency. This consistency and widely accepted interface between code and infrastructure enable tools like Infracost to parse the code to estimate the cost impact of code changes, best practices, tags, etc.

DevOps: DevOps is a set of practices that combines development (Dev) and IT operations (Ops). It aims to shorten the system development life cycle and provide continuous delivery with high software quality via automated processes.

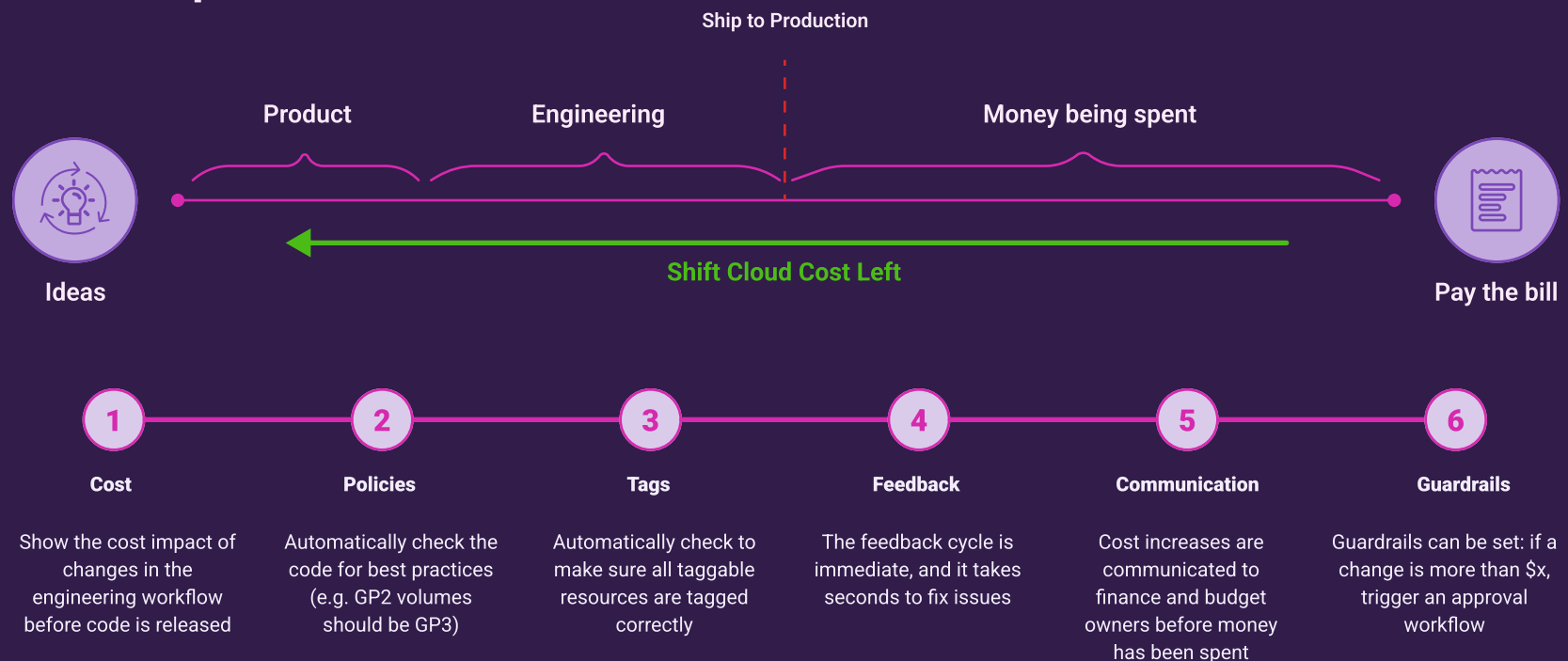
The development and adoption of these two fields have been the key to unlocking the shifting left of FinOps, and this is how Infracost has gained such wide adoption.

Shift Costs Left to Pre-Provisioning

The first win is to shift cloud costs left directly into the engineering workflow before resources have been provisioned. This will help the engineers understand how their choices will impact cloud costs before money has been spent.

Remember, the engineers who are the buyers of infrastructure have never been shown the cost of infrastructure changes upfront - they've never had a checkout screen before!

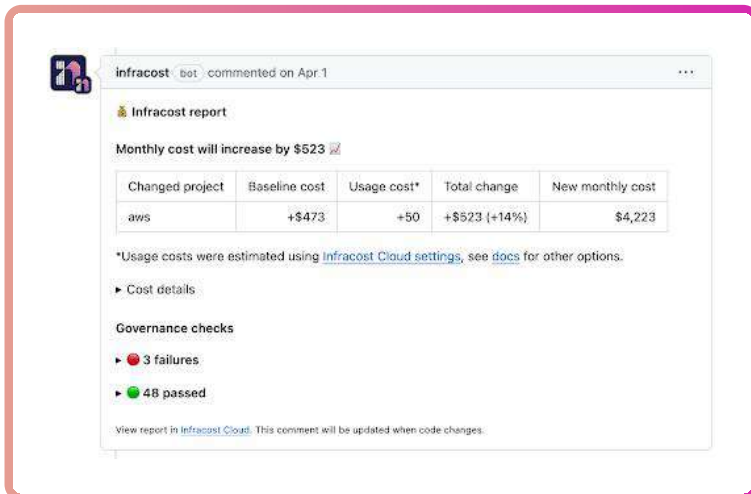
Shift FinOps Left



Shift Costs Left to Pre-Provisioning

Currently, Infracost tracks prices from Amazon Web Services, Microsoft Azure, and Google Compute Engine. From these providers alone, we track nearly 4 million price points. Estimating how every code change will impact cloud costs is impossible without having the right tools.

Infracost CI/CD was designed to address this problem. It's a very easy-to-deploy application that sits in your engineering workflow. Every time an engineer makes an infrastructure code change, Infracost posts a comment like 'This code change will increase your cloud costs by 20% next month.' The output also includes full details of all the resources the engineer adds, changes, or removes.



The screenshot shows a chat message from 'infracost bot' dated Apr 1. The message contains an 'Infracost report' with the following details:

- Monthly cost will increase by \$523
- Table with 5 columns: Changed project, Baseline cost, Usage cost*, Total change, New monthly cost.
- Footnote: *Usage costs were estimated using Infracost Cloud settings, see docs for other options.
- Cost details section.
- Governance checks section showing 3 failures and 48 passed.
- Footer: View report in Infracost Cloud. This comment will be updated when code changes.

Changed project	Baseline cost	Usage cost*	Total change	New monthly cost
aws	+\$473	+50	+\$523 (+14%)	\$4,223

THREE MAIN BENEFITS

1. SPEED UP ENGINEERS

This helps the engineer understand the costs of their chosen options without additional work or time spent going through cloud provider pricing pages. For example, they will see that provisioned IOPs on volumes are easy to request but relatively expensive.

2. KNOWLEDGE TRANSFER

This is where your other engineers review code for quality and security. Infracost now enables cost review. At this point, the more senior or experienced engineers transfer knowledge to the junior engineers. Examples we have seen are 'We don't need this big of an instance,' 'This is too much for this application,' or 'This setup makes it too expensive,' and 'If we did this x way, it would be cheaper.'

3. COST PREVENTION

Given that all this information is provided to the engineers before anything has gone to production, changes at this point will prevent these costs from going onto the cloud bills in the first place. Prevention is much better than trying to fix things after deployment.

Proactive Guardrails

Currently, the best practice for monitoring overspending is to set budget alerts through the cloud providers. Budget alerts work by looking at the current running bill, or the run rate, and forecasting it to the end of the month. If these bills exceed the amount you have budgeted for, an alert will be sent. The key issue with this is that after you get an alert, your budgets will break by default unless you interrupt your engineering teams to make changes to bring the current run rate back into budgets.

This is a challenging task for **two** reasons:

1. Your engineering team already has a lot of work to do on their sprints to deliver features, so you will be competing for sprint time. This means you will have to convince the engineering management and the product owners to prioritize FinOps tasks, but they have feature delivery goals and deadlines.
2. Analyzing how to reduce the bills and why they will break budgets takes a lot of time, so by the time the fix is found and released, it is probably already too late, and the task becomes containment and not prevention.

Proactive Guardrails

However, now that we can price out the cost impact of code changes before any money has been spent, we can also be proactive about budgets. Much like any other expense system, we can create a guardrail to monitor for cost increases coming from code changes, and if a code change is going to break our budgets, we can be alerted.

This is how Guardrails work in Infracost. The central FinOps team can set different business units, teams, or project thresholds.

All engineers can launch resources when needed; however, if one of these thresholds is exceeded, an alert is sent to the budget owner, and the engineer is told that this alert has been triggered.

The central FinOps team can also select to block the merging of the code and trigger an approval workflow for the change. This will notify the budget owner of who is making a change, what the change is for, and the price increase of the change. The budget owner can then discuss these changes with the engineers and either try to reduce the spending before it happens or set expectations that the budget needs to be increased due to the changes.

4. Thresholds

This guardrail will be triggered when the cost of a matching project inside a pull request exceeds all following thresholds. Cost thresholds are [currency-independent](#).

- Diff: Cost change
Trigger this guardrail when costs are increased by more than
- Diff: Cost change percentage
- Budget: New monthly cost

Example

This guardrail will trigger when cost change exceeds \$1000.

In the following example, the base branch cost of \$200 is increasing by \$100 or 50%, to a new monthly cost of \$300.

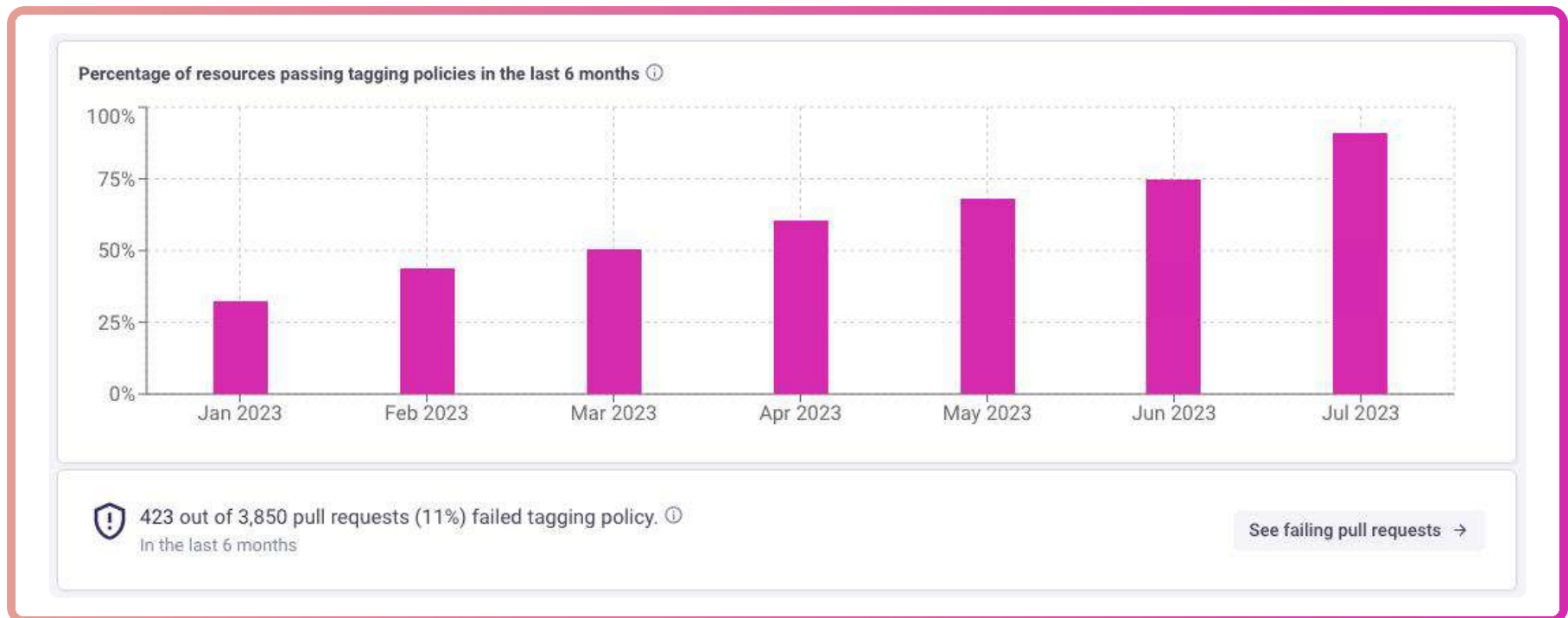
Pull request	Cost change	New monthly cost
Increase cluster size	\$100 (50%) ↑	\$300

Cloud Tags Checked in The Code

One of the tasks given to FinOps teams is to determine "who is using what"—that is, which teams, business units, products, etc., are spending the most on cloud resources. To accomplish this, they use tags. Tags are labels that all cloud resources should have and are key-value pairs. For example, a server could be tagged with "product=iPhoneApp," "environment=production," and "team=blueTeam." If resources are not tagged properly, you can't tell who uses what.

FinOps teams face challenges because current tools are reactive when detecting tagging issues. These tools begin by analyzing cloud bills and providing tag visibility from there. By definition, this means this data is only visible after the engineers have shipped their code to production; therefore, it is too late.

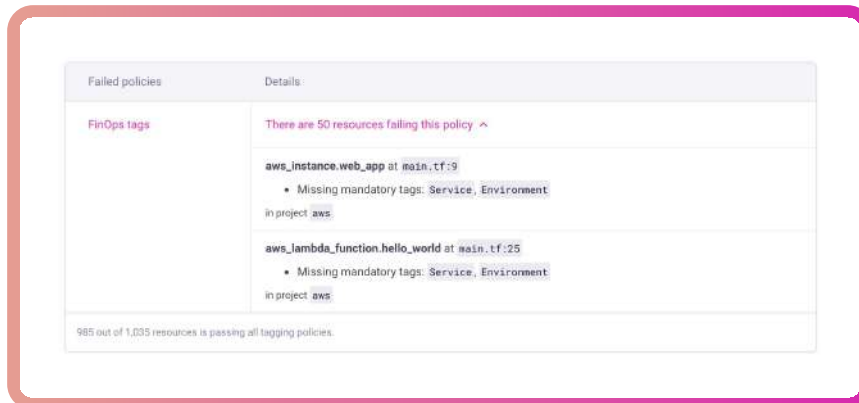
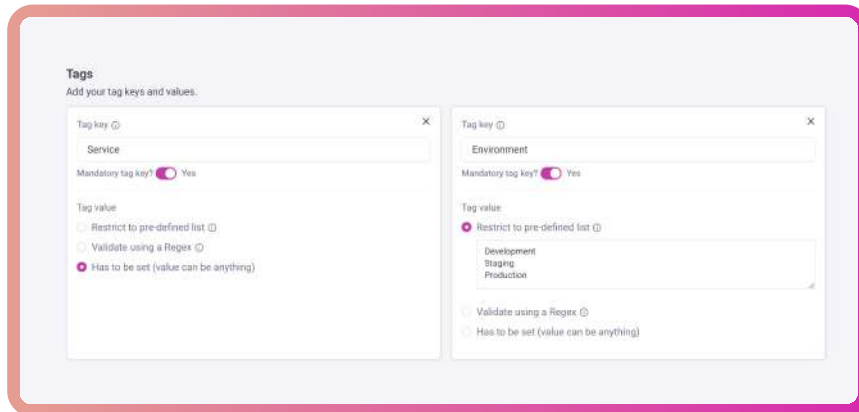
Now that we can scan the code before merging, we can also ensure that all resources are appropriately tagged.



Cloud Tags Checked in The Code

The central FinOps team can set their tagging policy in Infracost Cloud. They define what tags need to exist on resources and their values. There are three options for tagging policies: a pre-defined list that all tags' values must match, a regular expression (a sequence of characters that specifies a match pattern in text), or the tag must be present as long as it has a value and is not empty.

Infracost maintains a mapping of all cloud resources that can be tagged. If an engineer tries to add a resource that fails the tagging policy, they are told in their workflow. These tagging policies can also be enforced, meaning the code cannot be merged if the policy fails.



Failed policies	Details
FinOps tags	There are 50 resources failing this policy <ul style="list-style-type: none">aws_instance.web_app at main.tf:9<ul style="list-style-type: none">Missing mandatory tags: Service, Environmentin project awsaws_lambda_function.hello_world at main.tf:25<ul style="list-style-type: none">Missing mandatory tags: Service, Environmentin project aws
985 out of 1,035 resources is passing all tagging policies.	

There are **two** main benefits of doing this:

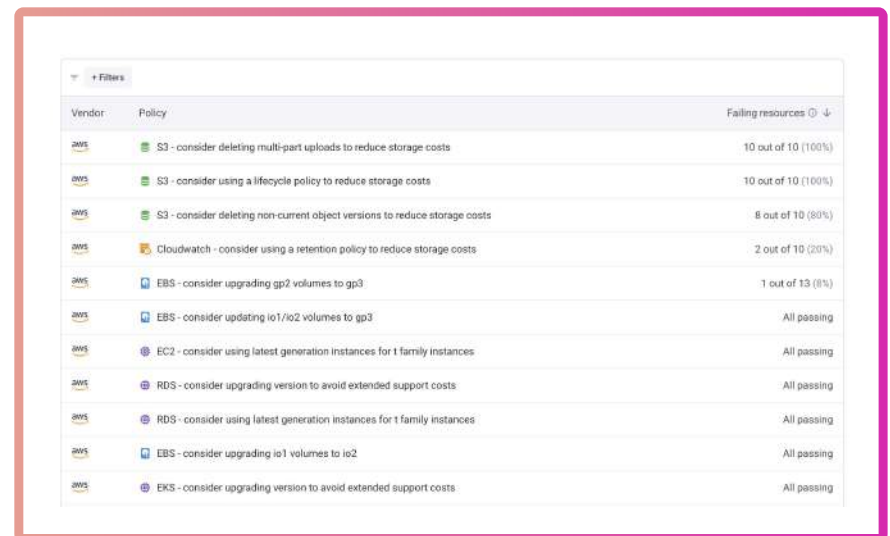
1. There will no longer be missing tags, typos in the values, or uppercase/lowercase issues within tags. This will help FinOps complete their task of knowing where money is being spent.
2. Engineers save a lot of rework. All resources that can be tagged will be checked, and the engineer will be told precisely what they need to fix. This means it takes them seconds to fix tagging issues vs. hours to do an end-to-end release to fix the pre-existing issues.

FinOps Policies Checked in The Code

Cloud providers are constantly updating their services with new features and versions. The FinOps foundation and Well-Architected frameworks provide many best practices on how to use services effectively. However, keeping track of these changes and recommendations can be a lot of work. It becomes even more complex when trying to determine which changes apply to which part of your codebase.

EXAMPLES OF SUCH POLICIES ARE:

1. AWS GP2 volumes should be GP3; io1 volumes should be io2.
2. AWS t2 instances should be t3; g2 should be g3; m2 should be m3.
3. AWS S3 should have lifecycle policies applied.
4. Azure NVv2 instances should be NVv3; Ls instances should be Lsv2.
5. Azure Storage accounts for blob storage should have lifecycle policies.
6. Azure SQL - Azure Hybrid Benefit should be looked at for SQL Server.
7. Google E2 instances offer up to 31% discount over N1.
8. Google Storage's unattached disks could be removed.

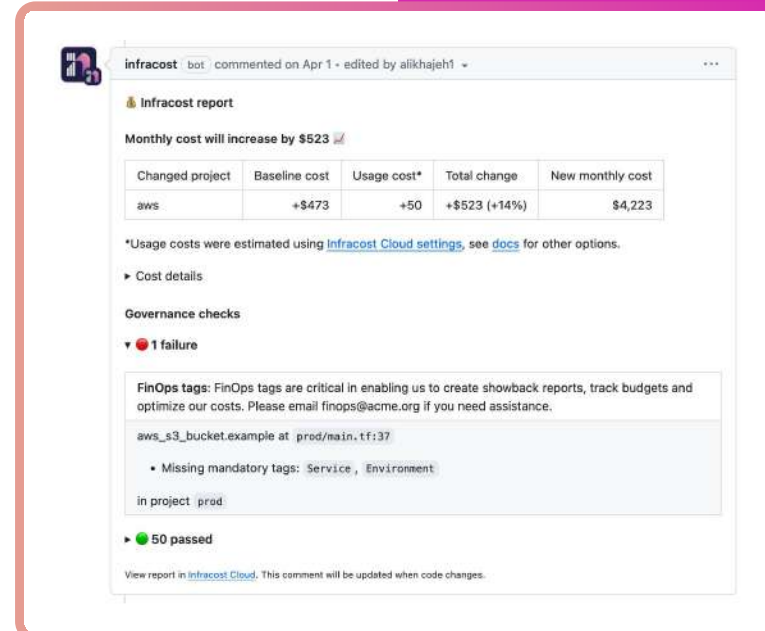


Vendor	Policy	Failing resources
AWS	S3 - consider deleting multi-part uploads to reduce storage costs	10 out of 10 (100%)
AWS	S3 - consider using a lifecycle policy to reduce storage costs	10 out of 10 (100%)
AWS	S3 - consider deleting non-current object versions to reduce storage costs	8 out of 10 (80%)
AWS	Cloudwatch - consider using a retention policy to reduce storage costs	2 out of 10 (20%)
AWS	EBS - consider upgrading gp2 volumes to gp3	1 out of 13 (8%)
AWS	EBS - consider updating io1/io2 volumes to gp3	All passing
AWS	EC2 - consider using latest generation instances for t family instances	All passing
AWS	RDS - consider upgrading version to avoid extended support costs	All passing
AWS	RDS - consider using latest generation instances for t family instances	All passing
AWS	EBS - consider upgrading io1 volumes to io2	All passing
AWS	EKS - consider upgrading version to avoid extended support costs	All passing

FinOps Policies Checked in The Code

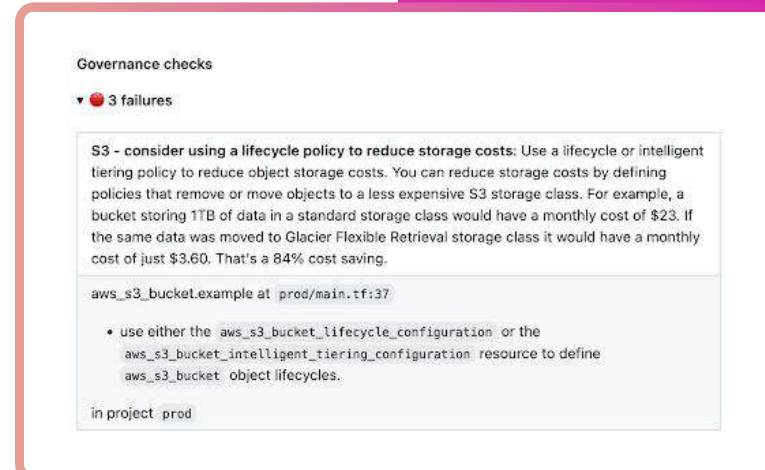
To simplify these policies, we have codified many best practice policies and offer these as defaults when you sign up for Infracost. First, we scan your current codebase and show you all the instances of your codebase using non-best practice resources. This enables you to create tickets with engineering to fix resources with all the exact code lines that need to be changed.

Next, we scan new code for every code change and inform the engineer directly in their workflow of how they can fix these issues. At this point, the code change takes seconds to fix. This not only helps remove a lot of issues from your code and bill but also saves a lot of rework from engineers having to fix and re-release code if the issue was caught after merging.



The screenshot shows a comment from 'infracost bot' on April 1, edited by 'alikhajeht1'. The report title is 'Infracost report' and states 'Monthly cost will increase by \$523'. A table shows the cost breakdown for the 'aws' project. Below the table, there is a note about usage cost estimation, a link to 'Cost details', and a section for 'Governance checks' with 1 failure. The failure details mention 'FinOps tags' and list missing mandatory tags 'Service' and 'Environment' for an AWS S3 bucket resource in the 'prod' project. At the bottom, it indicates '50 passed' and provides a link to view the report in Infracost Cloud.

Changed project	Baseline cost	Usage cost*	Total change	New monthly cost
aws	+\$473	+50	+\$523 (+14%)	\$4,223



The screenshot shows a 'Governance checks' section with 3 failures. The first failure is titled 'S3 - consider using a lifecycle policy to reduce storage costs'. The text explains that using a lifecycle or intelligent tiering policy can reduce storage costs. It provides an example: a bucket with 1TB of data in a standard storage class costs \$23, while moving it to Glacier Flexible Retrieval would save 84% to \$3.60. The failure details include the resource path 'aws_s3_bucket.example at prod/main.tf:37' and a list of suggestions: 'use either the aws_s3_bucket_lifecycle_configuration or the aws_s3_bucket_intelligent_tiering_configuration resource to define aws_s3_bucket object lifecycles.' The resource is located in the 'prod' project.

Inform Product Owners in Their Workflow

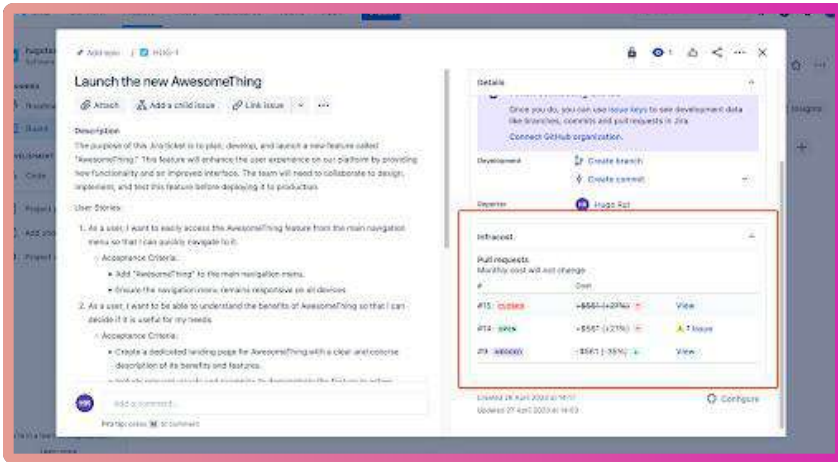
Earlier, we discussed the multiple roles that must be involved in FinOps. One of these roles is the product owner, who needs to understand how their feature requests will impact the cost of running the product or service.

Most change requests come from the business (product managers, product owners, or managers). These change requests are either changing features or adding new capabilities. Engineers will take these requirements and write the features. As part of that, they will also require new infrastructure or changes to existing infrastructure. These will have a cost impact. This cost impact needs to be fed back directly to the product managers so they are also aware of the cost impact of features.

Infracost has a direct integration with Jira. When an engineer sends a code change, Infracost automatically estimates the cost of the change. It then shows the cost impact to the engineer and pushes the cost impact directly into Jira under the issue the engineer is working on. This is linked automatically with GitHub, GitLab, or your source control system.

If multiple engineers make changes to the same Jira issue (e.g., a feature request, a cost reduction exercise, etc.), Infracost will sum up all the changes and push them to Jira.

Inform Product Owners in Their Workflow



1. Awareness of cost impact: In many companies, the Profit and Loss (P&L) of a product is owned by the product organization. This capability will help them see how different features and capabilities within their products will impact the P&L of their area.

2. Budget communication: Changes that break the budget are automatically communicated to the engineering team and the product organization. If a feature requires a bigger budget, the product team can set expectations with executives and management on the increased budget requirements.

There are **two** main benefits of informing the cost impact to product owners in their workflow:

Celebrate The Engineers

The FinOps Foundation has been conducting an annual report called "The State of FinOps" for three years now. In each report, FinOps practitioners reported that their biggest challenge was to persuade engineers to take action on FinOps tasks. The main problem behind this issue is that engineers are busy trying to deliver over-committed sprints. Additionally, when an engineer does not see a task as enjoyable or does not understand how it will add value to end-users, it gets mentally deprioritized.


This is similar to how you might stick a reminder on your wall to go to the gym, but eventually deprioritize it if you do not enjoy it.

The solution? Intrinsic motivation vs. extrinsic motivation. When an engineer is intrinsically motivated to fix issues, solve problems, and complete tasks, it becomes the task to complete to get away from other "boring" tasks.

One of the best ways to create intrinsic motivation is to gamify the tasks that must be completed. In Infracost, we have created a product section called **"Celebrating the engineers."** This shows the top people in two categories:

1. The engineers who have fixed the most tagging issues or FinOps policies.
2. The engineers who have reduced the cloud bills the most.

👏 Celebrating the engineers 🎉

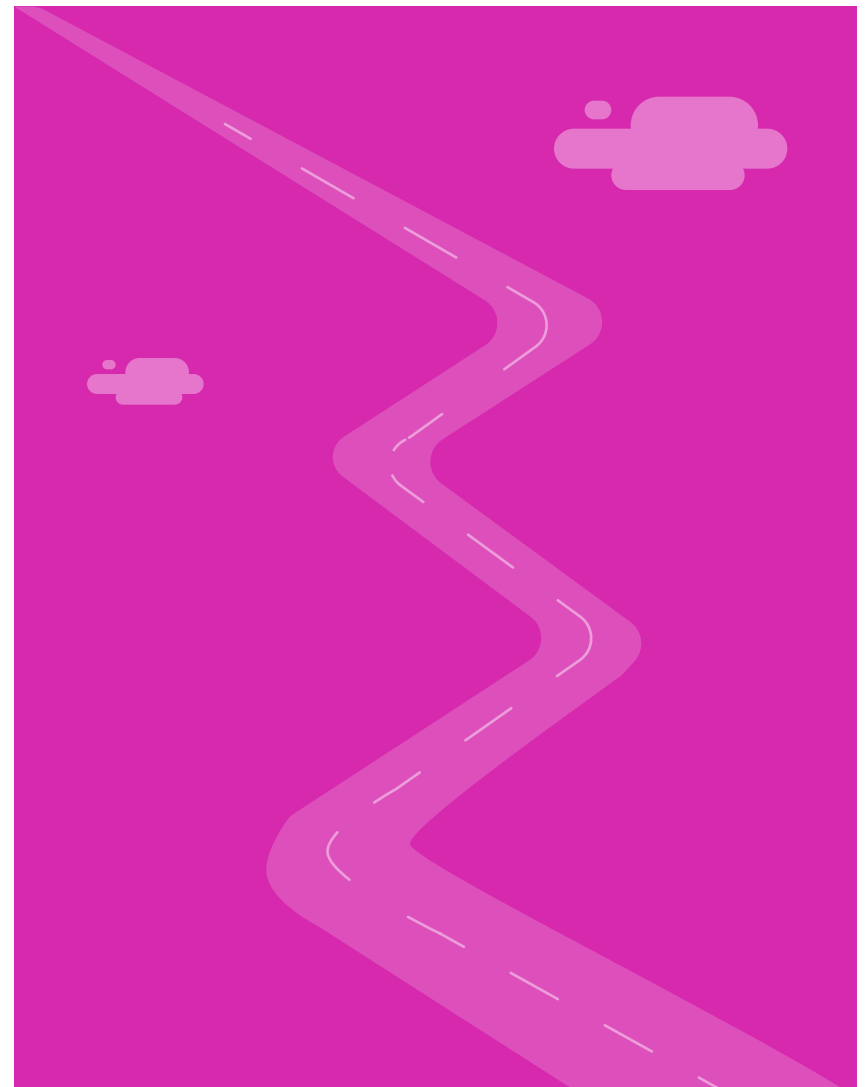
Governance heroes	Issues fixed	Cost heroes	Cost reductions
 Zeina Mohallel 🏆	74	 Priyanka Patel 🏆	\$40,210 /yr
 John Doe 🏆	69	 Alex Smith 🏆	\$37,094 /yr
 Scott McDonald 🏆	66	 Maciej Kowalski 🏆	\$32,384 /yr

A Real-World Case Study: Bourne Leisure's Journey to Proactive Cloud Cost Management with Infracost

Bourne Leisure, represented by Madoc Batters, Head of Cloud Centre of Excellence (CCoE), and Rich Young, Cloud Architect, is a testament to effective cloud management and cost optimization in a complex business landscape. The company, operating prominent UK holiday brands, faced the intricate challenge of managing a diverse cloud infrastructure across its vast array of services. This challenge was amplified by their holiday brands' varying needs and the leisure industry's dynamic nature.

At the heart of Bourne Leisure's strategy was a commitment to infrastructure-as-code, emphasizing Terraform and GitHub for all deployments. Madoc and Rich's focus on standardization, repeatability, and efficiency in cloud deployments underscores their forward-thinking technological approach. They recognized early on that managing cloud costs effectively was not just about technology but about creating a culture where every team member is empowered to make data-driven decisions.

Madoc and Rich's insights into the integration and impact of Infracost at Bourne Leisure highlight the tool's role in transforming their approach to FinOps and infrastructure optimization.



A Real-World Case Study: Bourne Leisure's Journey to Proactive Cloud Cost Management with Infracost

Challenges

Bourne Leisure encountered several challenges in their journey towards efficient cloud infrastructure management. Their primary challenges revolved around the complexities of maintaining consistency and efficiency in cloud operations across various brands within the company.

One significant challenge was the lack of standardization in cloud deployments. As Madoc pointed out, "If you spin up an AWS account and try and do it all in a console, there's no way you're going to get any level of standardization." This lack of standardization made management difficult and posed risks in terms of security and compliance.

Another critical issue was the focus on rapid deployment, often at the expense of cost optimization and compliance. Rich highlighted this concern, noting that developers are driven towards "time to delivery, not time to consistency, or time to security, or time to cost optimization." This approach often led to deployments that were not cost-effective or fully compliant with company policies. For example, previous generation instance types were used, or tags required for cost reports were missed or had typos.

Furthermore, the complexity of cloud pricing models added another layer of difficulty, making it challenging for the team to understand and manage cloud costs effectively. Madoc emphasized the difficulty of grasping cloud pricing, which consisted of "so many different services."

These challenges necessitated a solution that addressed cost management and encouraged a shift in the culture towards more efficient and compliant cloud infrastructure deployment.

A Real-World Case Study: Bourne Leisure's Journey to Proactive Cloud Cost Management with Infracost

Solution

The solution to Bourne Leisure's cloud management challenges was refined through a 'shift-left' approach with Infracost, enhancing sustainable and efficient cloud management. By integrating Infracost into CI/CD pipelines, Bourne Leisure achieved real-time feedback on cloud costs, which is pivotal for informed resource deployment decisions. This integration, emphasized by Madoc, enabled dynamic assessment of cost implications, aligning with company policies for cost-effectiveness and compliance.

Madoc emphasized the value of this integration, stating, "We've integrated it into our CI/CD pipelines. It's about getting that immediate feedback." This immediate insight allowed the cloud engineering team to dynamically assess the cost implications of their infrastructure decisions, ensuring cost-effectiveness and compliance with company policies.

Furthermore, Infracost's adoption fostered a cultural evolution within the engineering teams, embedding cost-awareness into development processes. Rich noted this cultural shift, highlighting the emphasis on preemptive cost considerations. This 'shift-left' strategy facilitated a proactive stance on cost management and ingrained a disciplined approach to cloud resource deployment, aligning technical initiatives with financial accountability and operational efficiency.

In summary, the solution to Bourne Leisure's challenges was not just a tool but a transformation in their approach to cloud infrastructure management, with Infracost at the core of this change.

A Real-World Case Study: Bourne Leisure's Journey to Proactive Cloud Cost Management with Infracost

The Infracost Impact

Integrating Infracost into Bourne Leisure's practices brought a transformative shift in cloud infrastructure management, notably through the 'shift-left' approach. By embedding Infracost directly into their CI/CD pipelines, Bourne Leisure's team gained early insights into cloud costs, enabling a proactive stance on managing expenses.

This new approach significantly shifted the company's cloud management culture. The engineering teams began to incorporate cost considerations into their decision-making process more thoroughly, enhancing their understanding of the financial implications of their cloud infrastructure choices. Infracost's real-time cost feedback was instrumental in this transformation. This integration allowed for immediate visibility of potential costs, facilitating more informed and cost-effective decision-making.

Madoc and Rich underscored Infracost's role as more than a tool; it became a partner, offering real-time feedback and fostering a relationship based on collaboration and mutual growth. This partnership was pivotal in standardizing deployments and ensuring cost efficiency across diverse cloud environments, marking Infracost as both an effective cost management solution and a catalyst for change within cloud engineering practices, aligning technical decisions with financial awareness and responsibility.

A Real-World Case Study: Bourne Leisure's Journey to Proactive Cloud Cost Management with Infracost

Results

Implementing Infracost at Bourne Leisure led to significant results, as Madoc and Rich elaborated. The tool's integration into their cloud infrastructure significantly enhanced their cost management and optimization strategies

The overall impact of Infracost at Bourne Leisure was profound, demonstrating its effectiveness as a comprehensive cloud cost management and optimization solution.

1. COST PREVENTION

A major achievement was the prevention of around 15% in cloud costs. This was a direct result of engineers gaining visibility into potential costs before deploying changes, enabling them to make more optimal decisions.

2. ISSUE DETECTION AND BEST PRACTICE IMPLEMENTATION

Infracost's capabilities extended beyond cost prevention. It played a vital role in detecting over 2,000 issues across Bourne Leisure's code repositories. These issues covered critical areas such as tagging, compute, databases, block storage, object storage, and logging, aligning with FinOps best practices and policies.

3. TIME SAVINGS AND ENHANCED DECISION-MAKING

Infracost was not just a financial tool; it was transformative in saving significant time for engineering teams. This efficiency boost allowed the teams to focus more on strategic initiatives, enhancing business value. The tool enabled data-driven decision-making, aligning technical actions with Bourne Leisure's broader business objectives.

FinOps Benefits: Better Outcomes For Companies And Engineers

Cost Prevention And Savings

Infracost's proactive approach allows companies to foresee and adjust their cloud usage before costs incur, preventing budget overruns. This pre-emptive strategy has demonstrated the potential to cut cloud expenses by notable margins, offering a dual advantage of cost prevention and tangible savings.

Saving Engineering Time

In software engineering, if you catch issues early in the development process, they are quick and easy to fix. The longer you wait to resolve issues, as code goes to production and later, as the context of the code is lost in the engineer's mind, the harder and longer it becomes to fix.

Infracost is proactive, meaning that when an issue is caught, it takes seconds to fix versus hours if it is caught after it has gone to production. This time savings across hundreds or thousands of engineers is significant.

Cost-Efficient Code

With tools like Infracost, engineers can write cost-efficient code from the outset. This practice prevents cloud costs and elevates the engineer's role from a code creator to a cost-efficient solutions architect. It encourages a mindset where cost optimization is a hallmark of excellent engineering.

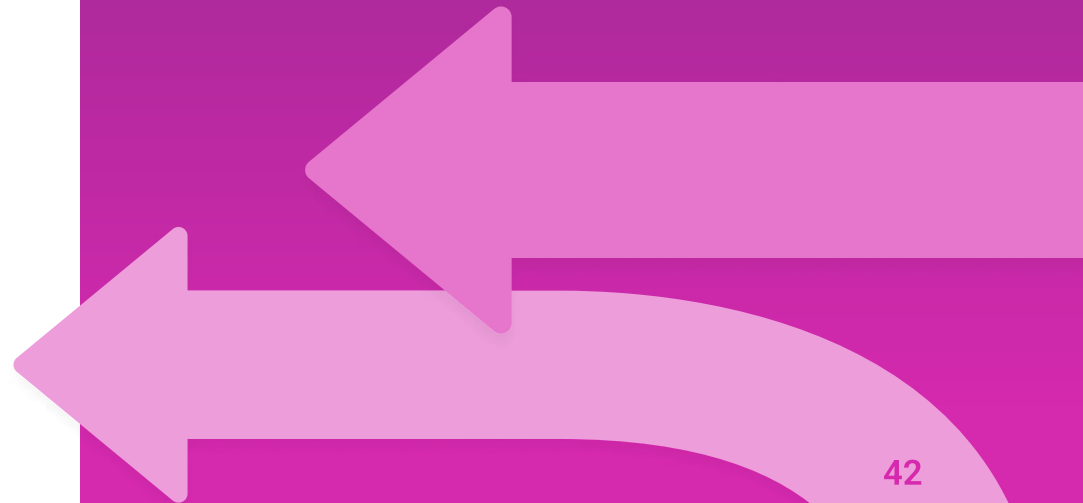
FinOps Policies Being Followed

Infracost shows engineers how to improve their code by following well-architected frameworks and FinOps policies. Integrating these directly into the engineering workflow allows engineers to quickly ensure their code follows these policies, such as ensuring all resources use the latest generation of storage and that all data storage has retention policies.

The Key to Success: Shifting FinOps Left

As we conclude, it's evident that the key to FinOps's success is to empower engineers. When engineers integrate cost considerations and FinOps policies early in the development process, they play a pivotal role in driving cost-efficient and sustainable cloud practices. This empowerment fosters a culture of accountability and innovation and aligns technical decisions with broader business goals.

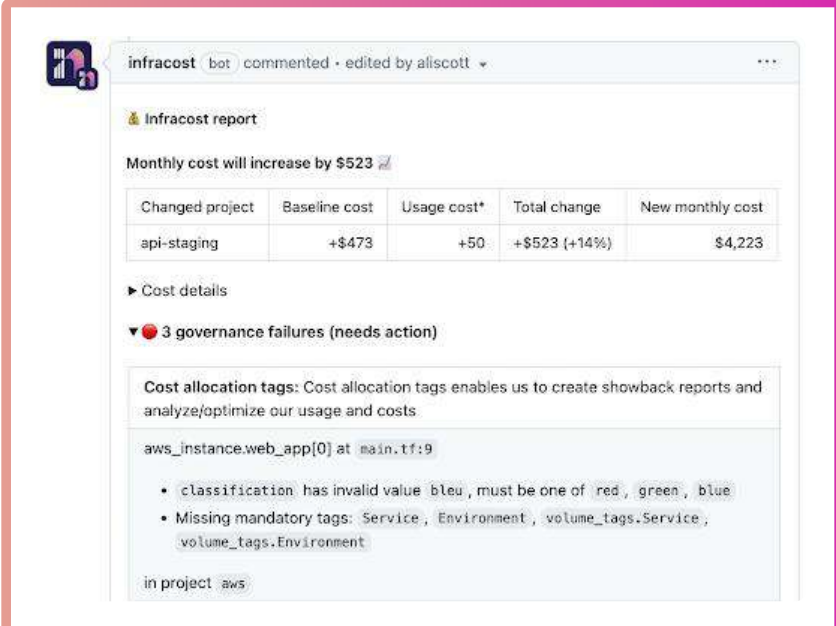
Ultimately, by shifting FinOps left, we optimize costs and enable engineers to design their organization's technological and financial future proactively.



The Key to Success: Shifting FinOps Left

Infracost is the first and the industry standard to shift FinOps left directly into the engineering workflow. It is used by over 3,000 companies, including some of the world's largest cloud deployments.

Infracost sits in your engineering workflows (GitHub, GitLab, Azure DevOps repos, BitBucket, etc.). When an Infrastructure as Code (IaC) change is made, it automatically posts a comment with the cost impact of the change into their workflow. It also scans all resources to ensure they abide by the FinOps tagging policy set by the central FinOps team. It also ensures the code follows best practices from the well-architected framework and FinOps policies.



The screenshot shows a comment from 'infracost bot' in a code repository. The comment contains an 'Infracost report' with a warning icon. It states 'Monthly cost will increase by \$523'. Below this is a table with columns: 'Changed project', 'Baseline cost', 'Usage cost*', 'Total change', and 'New monthly cost'. The table has one row for 'api-staging' with values: '+\$473', '+50', '+\$523 (+14%)', and '\$4,223'. Below the table are sections for 'Cost details', '3 governance failures (needs action)', and 'Cost allocation tags'. The 'Cost allocation tags' section includes a code snippet for 'aws_instance.web_app[0]' with two bullet points: 'classification: has invalid value bleu, must be one of red, green, blue' and 'Missing mandatory tags: Service, Environment, volume_tags.Service, volume_tags.Environment'. The code is shown in a light blue box with syntax highlighting.

Changed project	Baseline cost	Usage cost*	Total change	New monthly cost
api-staging	+\$473	+50	+\$523 (+14%)	\$4,223

Customers gain **two** main benefits by shifting FinOps lefts:

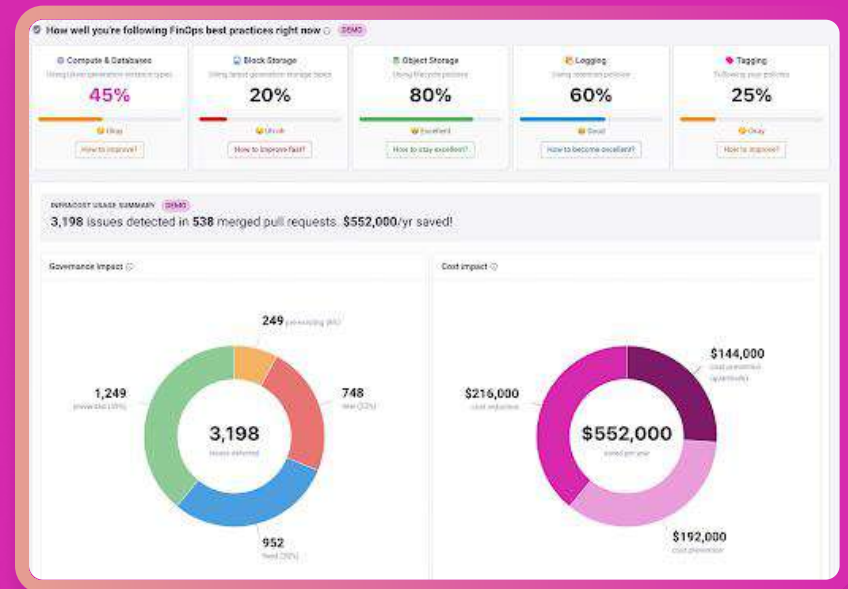
1. Fixing issues in the engineering workflow and preventing them from entering production in the first place can save engineering time. This reduces a lot of rework and makes the engineers much more productive and efficient.

2. Prevent cost waste from entering their cloud bills in the first place. You prevent waste from the start by catching out-of-budget and overspending before money has been spent.



Start Using Infracost Today

As you use Infracost, it will show how many issues were prevented as well as how much money was prevented from going into your bill in the first place:



It's easy to set up and does not require a connection to your cloud provider accounts. Ask your DevOps or Platform engineering teams to set up Infracost now by going to infracost.io and starting with a free account.

[Start your free trial](#)