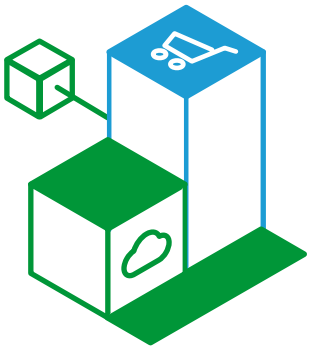
NGINX
Part of F5

The Masterguide to IT Agility

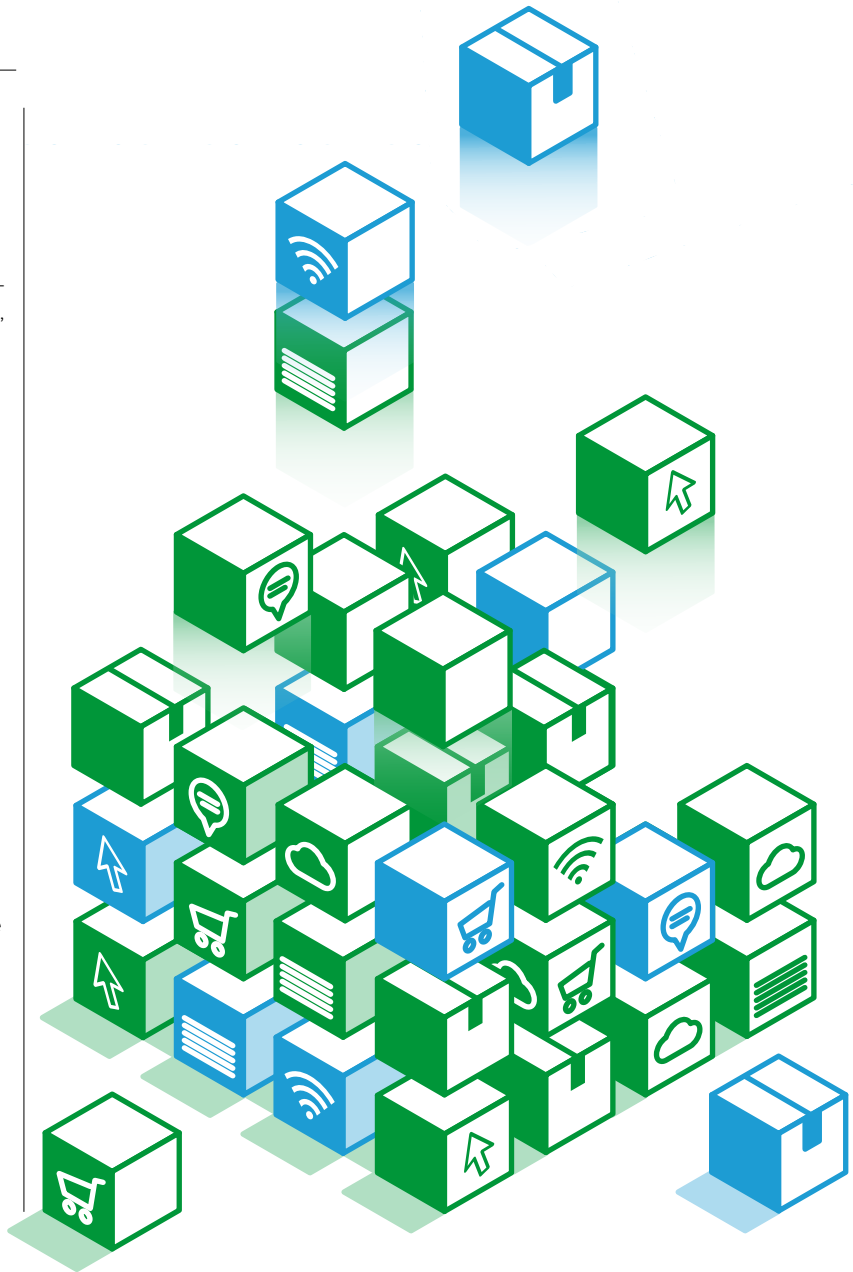
2020

Contents



The What, Why, and How of Business Agility

Traditionally, businesses have bolted on, refined, added, and subtracted code to systems infrastructure. Building from this foundation was historically sensible, but it's now hampering the rapid delivery of services



The way apps were conventionally designed can be thought of as retrieving services from a library: a two-way communication where the application searches the server-side monolith for what it needs and then delivers this back. Services are tightly coupled to centralised infrastructure and, if changes are made, a whole new library must be deployed.

Imagine, instead a constellation of nimble components, each of them performing very specific tasks, a decentralised army of microservices, fine-tuned to do their individual jobs, but all working together.

This mode of thinking, which positions developers at the centre of an organisation, is not new – agile charts back to the turn of the millennium – but now there's more documentation on hand and better, mature tooling, so it's much simpler to introduce processes into the pre-existing tangle of technology businesses already have in place.

Pioneered by Silicon Valley, microservices-led, developer-centric models help businesses move fast, making use of application programming interfaces (APIs), which enable software components to talk to each other, to co-ordinate all these moving parts. Ultimately, the goal is to automate laborious manual tasks that previously burdened teams, instead having all your infrastructure communicating efficiently, securely, automatically.

This is far from a technology problem alone. A major challenge is the social component, which can mean reconstituting the very DNA of an organisation.

In an ideal world, there'd be unanimous executive buy-in, infinite resources, and

We still need to standardise; you still need to get some level of complexity out of the system, but in a way that doesn't force your application to be tightly coupled to it

a talent pool unaffected by the skills gap. In the real world, people are protective of their patches. Teams butt heads. Conservatively minded leaders are reticent to throw resources at initiatives they may mistake for pie-in-the-sky thinking.

Business realities

That's before considering other business realities: most run legacy systems and are unable to launch from scratch with a coveted greenfield scenario. Many are fettered by the fact they're already successful businesses, with proven models that evolved over time to encompass a broad range of technology.

But all these crevasses can be crossed and businesses can make headway. Firms needn't commit vast sums from the get-go, but instead work incrementally and strategise how best to move towards desired outcomes.

There's been no better time to commit to this developer-centric model. Much of the technology is no longer experimental, with readily available community expertise on hand, particularly in the fast-moving world of open source, where some of the most intriguing developments are occurring.

Containers, for example, where applications are bundled into self-contained packages that can be spun up or down as necessary, are better supported. The Google-backed orchestration platform Kubernetes, which organises containers and allows them to run in multiple places at once, may have once intimidated, but is now much more mature.

The real-world competitive factor of open source is evidenced by F5's 2020 *State of Application Services* Report, which notes that while network automation from proprietary vendors remain the tools of choice, open source and continuous integration and delivery, or CI/CD, tooling is catching up quickly, with open-source automation server Jenkins having captured a massive 18% of the network automation userbase to date.

A boon from open source's quickly rising popularity in production is its central tenets necessitate moving away from the rigid world of enormous contracts with single vendors.

Creative and untethered

Developers wish to be creative and untethered, so being closely tied to underlying infrastructure, as they historically have been due to a desire to standardise, means tying developers' arms behind their backs.

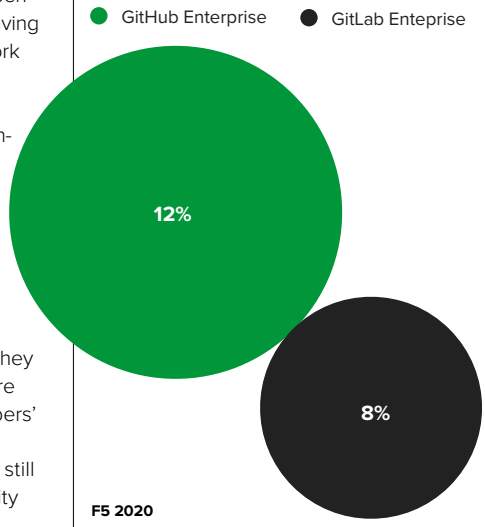
"We still need to standardise; you still need to get some level of complexity out of the system, but in a way that doesn't force your application to be tightly coupled to it," says Rob Whiteley, Vice President of Marketing at application services specialist F5. "This has to be an architectural principle: how do I create an abstraction layer that sits below my infrastructure so I'm able to standardise on the different services my developers need, but doing it in a way that isn't binding them to a particular application?"

The connective tissue to all these components are gateways and control points. API gateways manage the traffic flow and security of communications, while controllers such as Ingress administer the same functions, but in and out of the Kubernetes cluster. All these elements are specialised for the environments they oversee, and more than 80% of organisations with high API call volumes have deployed Ingress control on-premise, falling only to 67% on public cloud.

Imagine a sports stadium, says Whiteley, up and running for the first ever time. Without the various ticketing booths, stewards, signage, and checkpoints – the gateways and control points – footfall would be chaotic, with attendees ambling through the grounds in disarray. Perimeter control points are needed to prevent someone from wandering in from the street. Just as internal control points are needed to guide fans and provide them with a good experience. These layers and controllers, then, secure and make sense of all this traffic.

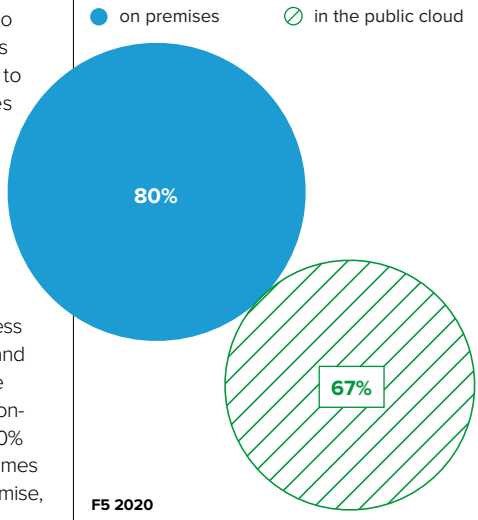
Wrestling with new technologies, and the mindset necessary to adopt them, may encourage a sense of inertia.

Respository usage



F5 2020

Organisations with high API call volumes that have deployed Ingress control



F5 2020

But there are clear routes towards reorienting your organisation to include all of these components, with tried-and-tested ways to increase alignment of teams.

If the ultimate aim of digital transformation is to move quickly, it follows that all strands of the operational model should match.

For your efforts, the rewards will be richly felt, releasing new services to customers in hours instead of weeks or months, and an organisation that's able to realign itself quickly based on business need. Next, we'll touch on how to start laying the foundation today to reap the benefits tomorrow.

Automate to Win: Your Guide to Achieving the Developer Mindset

Selecting the most appropriate way to design infrastructure automation for your developers is a challenge, but help is at hand

A good place to start is by taking stock of current capabilities on one hand, goals on the other, and mapping out a route for delivery.

An executive sponsor is enormously helpful; communicating benefits to the board, hearing and addressing their concerns, assists in opening a dialogue and putting together a framework for success.

Figure out your risk appetite and understand your teams' skillset: there's little use ploughing forward on a project that's utterly mismatched with your organisation's technical know-how without first taking steps to get up to speed, whether that's in upskilling or narrowing the scope of a project.

Without first thinking strategically, businesses risk running fake modernisation initiatives, where changes made are merely superficial, giving the impression of transformation, but without actually delivering on meaningful metrics such as speed. Transparently assessing capacity is crucial to avoid mis-selling a project, risking disappointment later.

Before a proof of concept is anywhere close to production, teams will want to lay out what their organisational model and technology stack needs to look like, to carefully craft the production pipeline.

There are endless combinations and no right answers; it may be a mixture of experimentation and working backwards from desired outcomes that gets you there.

It's entirely appropriate to look at the pipeline you want to build, and then use that as a criteria for selecting the most appropriate way to design automation in, says Liam Crilly, Director for Product Management at F5.

"There's so many competing inputs, so many variations even based on whether you're doing one specific public cloud vendor, or multi-cloud, or going private cloud," he says. "Those create a huge variety of deployment options, so there's not a right way or a wrong way."

Cross-functional team

A solid grounding could encompass a core unit consisting of a cross-functional team, covering design, implementation, testing, and security, all in one.

At the same time, hash out a combination of software tools that will assist in actually delivering the code.

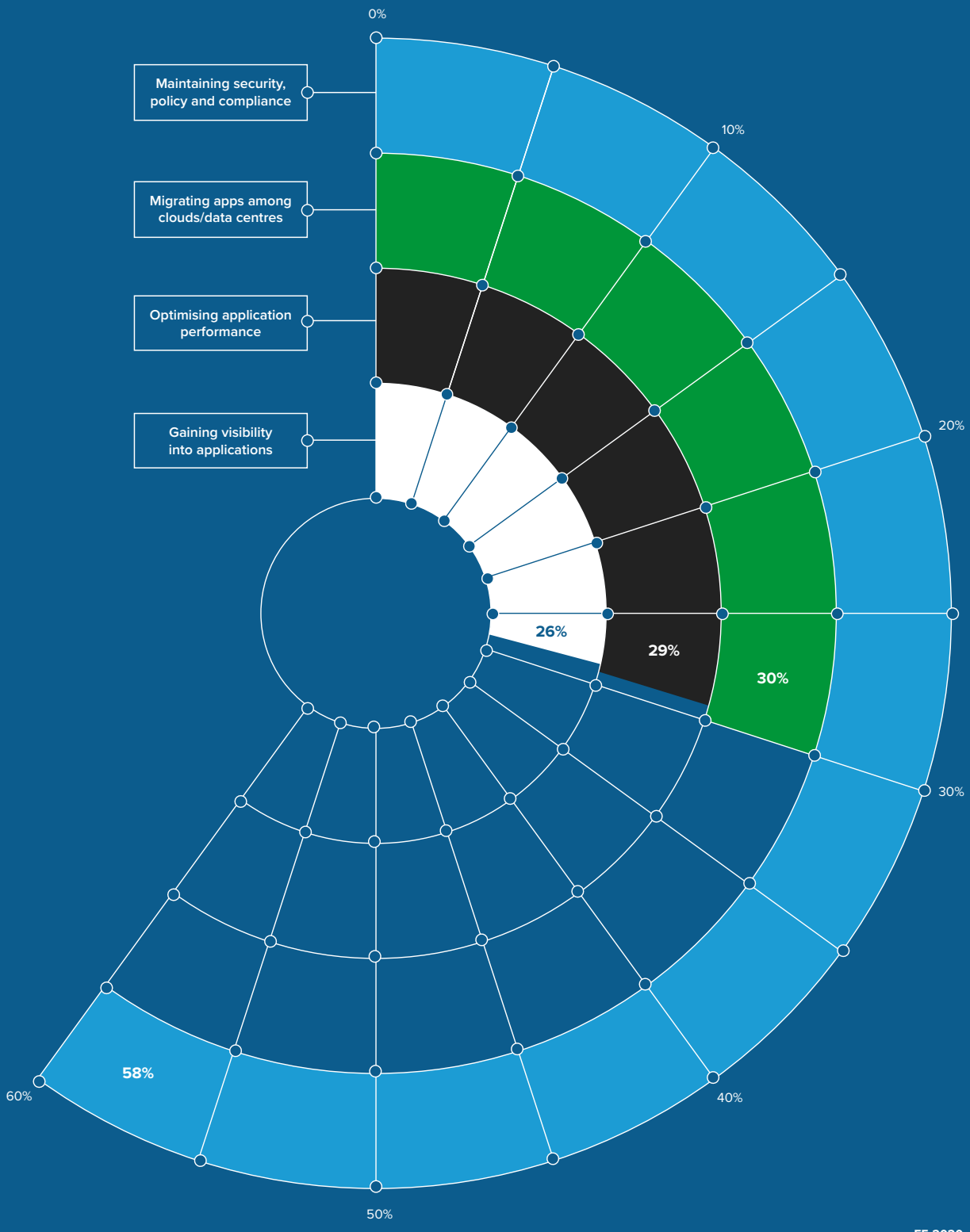
Most likely, teams will start with a set of core building blocks: one or two cloud providers, Docker for containers, orchestration platform Kubernetes, and an automation provisioning tool such as Ansible. The challenge is to ensure all these components and layers talk to each other efficiently, reliably, and securely. Key to achieving speed is automating as many processes as possible, enabling moving more swiftly, but without losing sight of quality.

There are endless combinations and no right answers; it may be a mixture of experimentation and working backwards from desired outcomes that gets you there

Whatever systems are already in place may not be the most compatible, so building an environment where technologies play nicely with one another is essential. Controllers and gateways will be essential to manage traffic flow, authentication, and security.

Getting security and network operators on board from the get-go is of vital importance. The last thing a project needs is to make headway only to find it nixed by security at the eleventh hour.

Parts of managing applications in a multi-cloud environment that organisations find the most challenging, frustrating or difficult



F5 2020

"Before I would even advocate running a single line of code or doing a proof of concept, I want to put in place a modern, microservices-friendly, DevOps-friendly, continuous integration and delivery pipeline, automation tests, frameworks for these things," says Crilly.

"They'll want to be in place so, as soon as you start writing code, there is a feedback loop and something that will carry you forward, rather than trying to work these processes into existing systems."

Automation tooling agenda

Consider an agenda you may need for your automation tooling or your existing environment. Along the way, ask questions like: what is the runtime environment? How do I run code review? Is my delivery vehicle a container? Is it orchestrated by Kubernetes? Are developers in the same physical location?

Thinking about questions like these will help towards running an end-to-end analysis, ensuring every note on the pipeline is hit, as well as identifying potential pain-points from planning through to delivery.

Consider introducing a whole new tech stack that will assist in metric collection.

"The problem of being distributed extends itself to observability and traceability," says Crilly, adding it will be worth looking at monitoring systems, such as Prometheus, to measure progress.

Automation tooling has progressed leaps and bounds, requiring far less customisation to stick the various parts together. Welcome news, since after the cloud, businesses note that automation and orchestration are the most strategically important technology trends in the coming two to five years, according to F5's 2020 *State of Application Services* Report.

However, it's worth ensuring that whatever the investment, it's not tightly coupled to any single provider, with applications spread across infrastructures in case of outages.

Of course, businesses will have to weigh up whether the immediate wholesale sunsetting of legacy environments and moving everything over to a greenfield, with the initial investment it entails, will be feasible.

Older systems

Many businesses will have a large number of customers running on these older systems and that means it's really hard to move from those in a short time, says Dr. Danjue Li, Sr. Director of Incubation Engineering, Product Research and Incubation (PRI) at Equinix.

"In that case, having interoperability capabilities with that existing system is extremely important," she says. "You have to run that brownfield at the same time and then during the process you will start to have new users served by the greenfield. There will be tangible and intangible business values." Organisations will have to look at everything together and make decisions holistically. "There's no one-size-fits-all solution," Li adds.

African Bank Technical Architect Fintan Wilson echoes this advice, pointing out teams should try to be as clear as possible on what the base environment is going to look like.

"I don't want to say people should lock down in terms of technologies, but you should have an idea of how you're going to framework your deployments," he says. "Once you're over that hurdle, you can provide the framework as a set of guidelines for developers. As long as they stay within this, you should be able to build a fairly robust environment."

African Bank, which successfully relaunched in 2016, was particularly stringent on these guidelines at first and over time relaxed some of the restrictions.

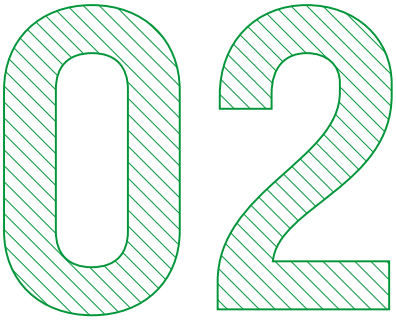
"The second leg is to make sure that once you define those frameworks, make sure people are onboarded with the journey; a lot of the time, people struggle to understand why something was chosen a specific way. If you take a bit of time, and take them through the thought process there, they're typically much more accepting and willing to get involved in trying to help," says Wilson.

Cost is inescapably a factor. Li suggests teams starting out should opt for technologies that provide flexible cost models, so they can proceed without having to commit to mammoth upfront payments.

"Whether it's a large company, a team within a large company, or a startup, it's critical to always be sensitive about the cost of building and then pick a technology that'll give you flexibility to have better control over that cost of building," she says. "Then scale that cost as you scale your business." ■

Technology trends that will be strategically important for businesses over the next 2-5 years

Cloud



Automation and orchestration

Big data analytics

Winning Buy-in

Now that you have a plan, you need to secure buy-in. Fortunately, the benefits of a developer-centric model are strong vantage points to pitch from; moving faster, better, and more efficiently should be an easy sell. The focus should be on elucidating, in language that resonates with decision-makers, how your roadmap will help get you there.

The high-risk, high-reward startup ecosystem is one route. Picture a business unit that's allowed to operate outside conventional strictures, unrestrained by processes and therefore freer, more flexible. Standing in the place of an investor, the C-suite sets resources such as time and budget so this unit can focus on its goals.

"What I don't want my developers doing is spending a lot of time coding low-value services; things that are problems, which have been solved time and time again: how to authenticate a user, how to put things in a shopping cart, how to handle some of the basic security mechanisms," says F5's Vice President of Marketing Rob Whiteley. This chimes with F5's 2020 *State of Application Services* Report's finding that in only 10% of organisations are developers the role primarily responsible for deploying and operating application services in the public cloud.

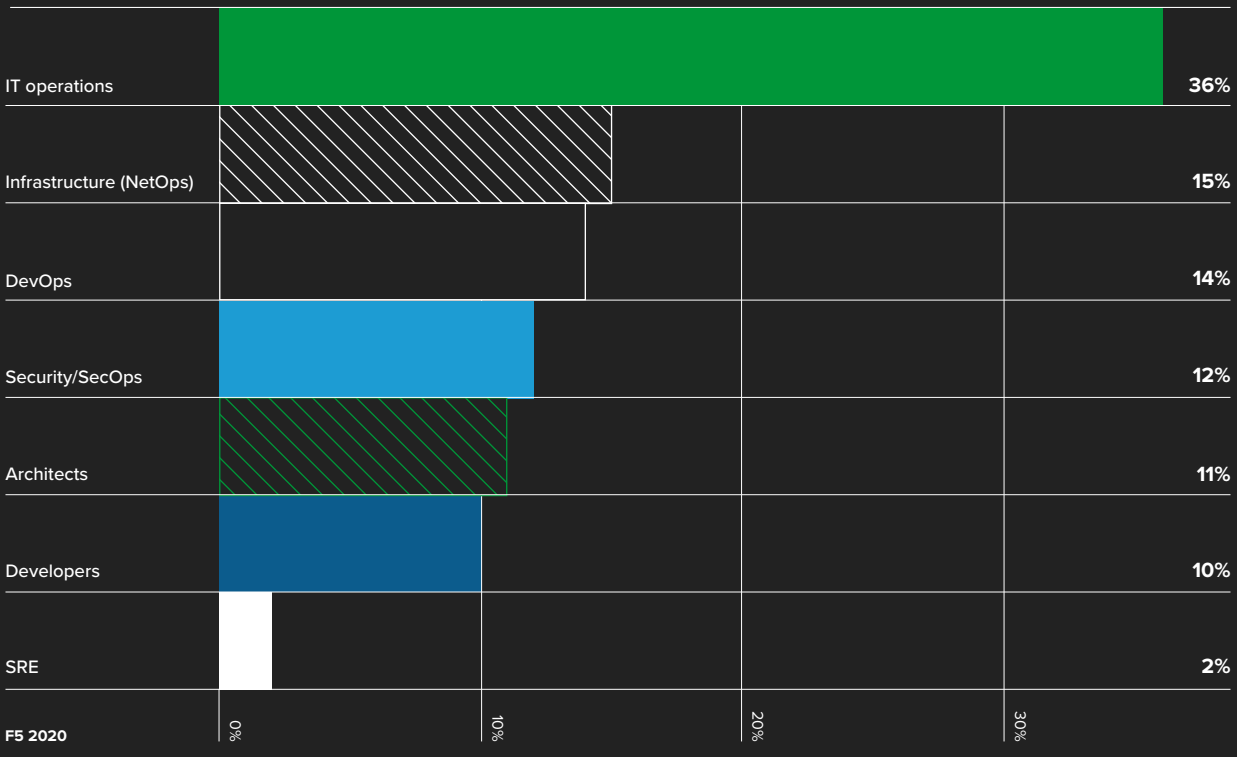
By clearly establishing goals and proposing a timeline, this model should help reassure executives there are controls in place.

The usual rules won't apply: criteria such as rate of customer acquisition take time, something that's in short supply for a limited pilot, so assessing what you're trying to do, then making this measurable, will help. Just as startups face continuous assessment, and must re-evaluate goals as they grow, internal projects should be under similar scrutiny.

"It's not something the C-suite should be afraid of," says Owen Garrett, Vice President of Product Management at F5. "They shouldn't think it's a case of engineers taking a step away from their day job, playing with new technology without a goal or cost controls."

There's give and take. An environment where the developer is at the heart of strategy, where teams are free to work with the tooling that suits them, and the sense of purpose from contributing to open-source communities, all reflect well on the organisation and are a powerful recruiting tool, Garrett concludes.

Roles within organisations that are primarily responsible for deploying and operating application services in the public cloud

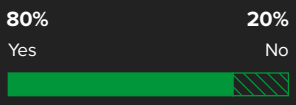


Empowering agility

As businesses of all sizes look to accelerate their digital transformation, applications are the engines powering the digital economy. What is the current outlook for application services, and how are they enabling organisations to adapt to new realities and hard-wire flexibility into their essence?

The vast majority of businesses are undergoing digital transformation...

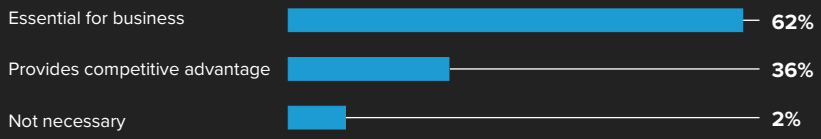
Organisations executing on digital transformation



F5 2020

...and applications are essential for most companies

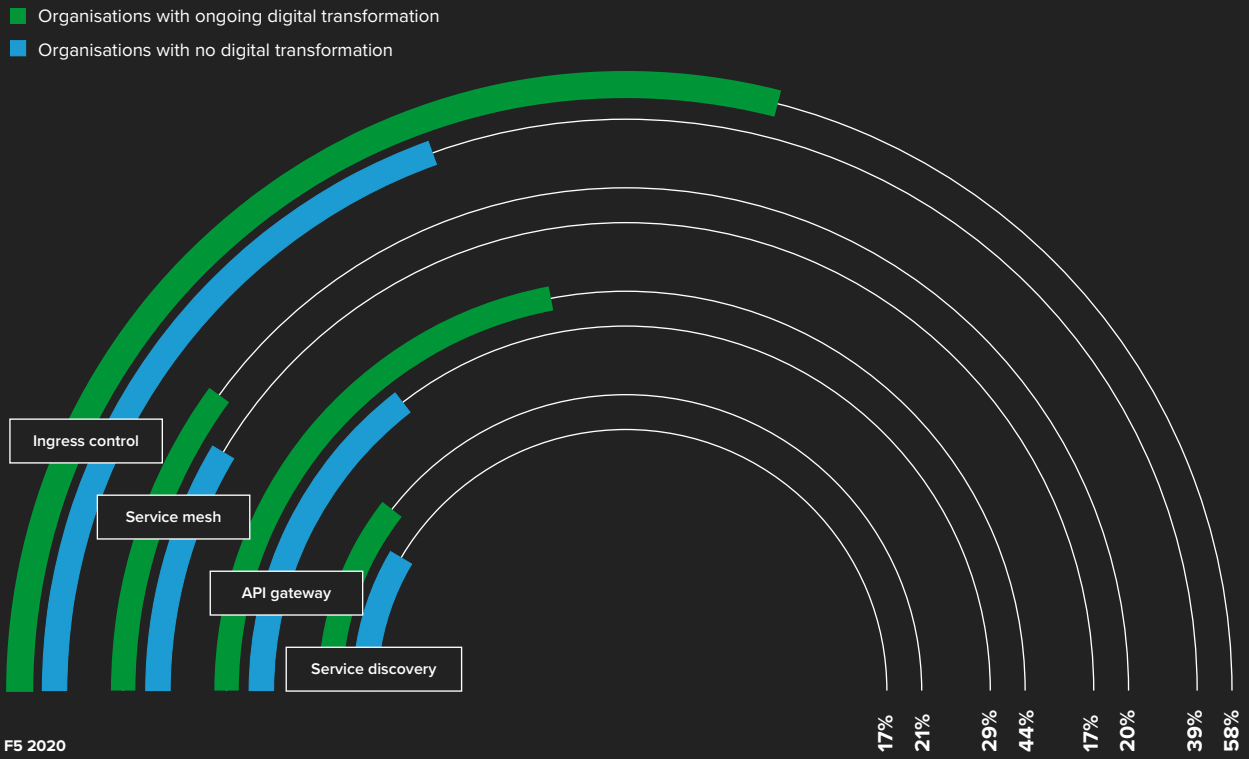
How organisations view/treat their application portfolio



F5 2020

Organisations undergoing digital transformation are more likely to deploy modern app architectures and app services at higher rates

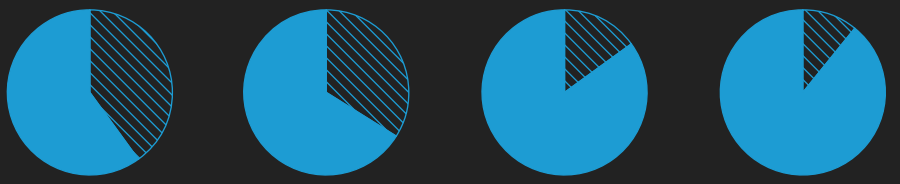
Application services being deployed in either on-premises data centre/private cloud or public cloud



F5 2020

With a diverse range being applied, no single application architecture has a clear majority

Applications currently being deployed



40% Three-tier web apps & mobile

34% Client-server

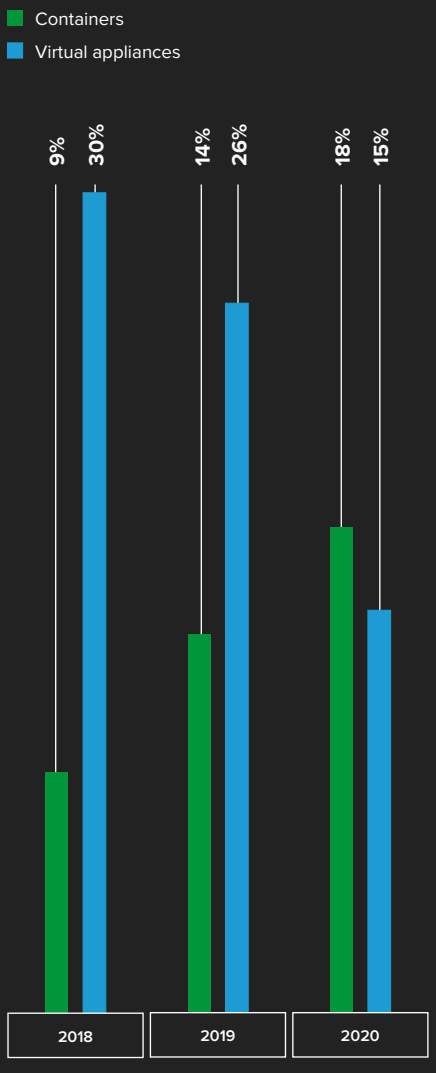
15% Cloud native/ microservices

11% Monoliths mainframes

F5 2020

There is a rapidly growing preference for containers over virtual appliances for application services

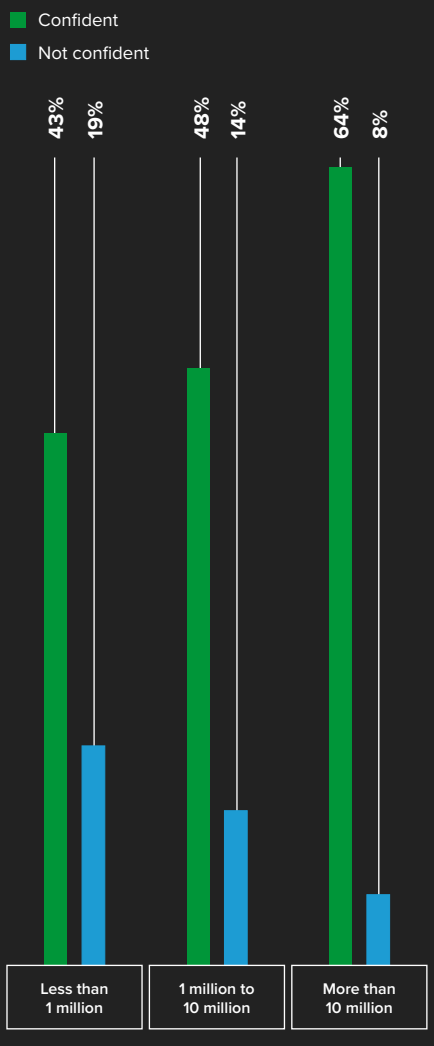
Preferred form factor for on premises application services



F5 2020

With API security a key challenge, API practice maturity is intrisically linked to an organisation's confidence to withstand an attack

Confidence in company's ability to withstand an attack against its APIs (by volume of API calls per month)



69%

of organisations are using 10 or more application services

F5 2020

77%

of containers are run on Kubernetes

Sysdig 2019

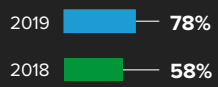
50%

growth in the number of organisations using Kubernetes as their container orchestrator from November 2018 to July 2019

Dzone 2019

Use of Kubernetes in production has soared year-on-year, with this trend likely to continue

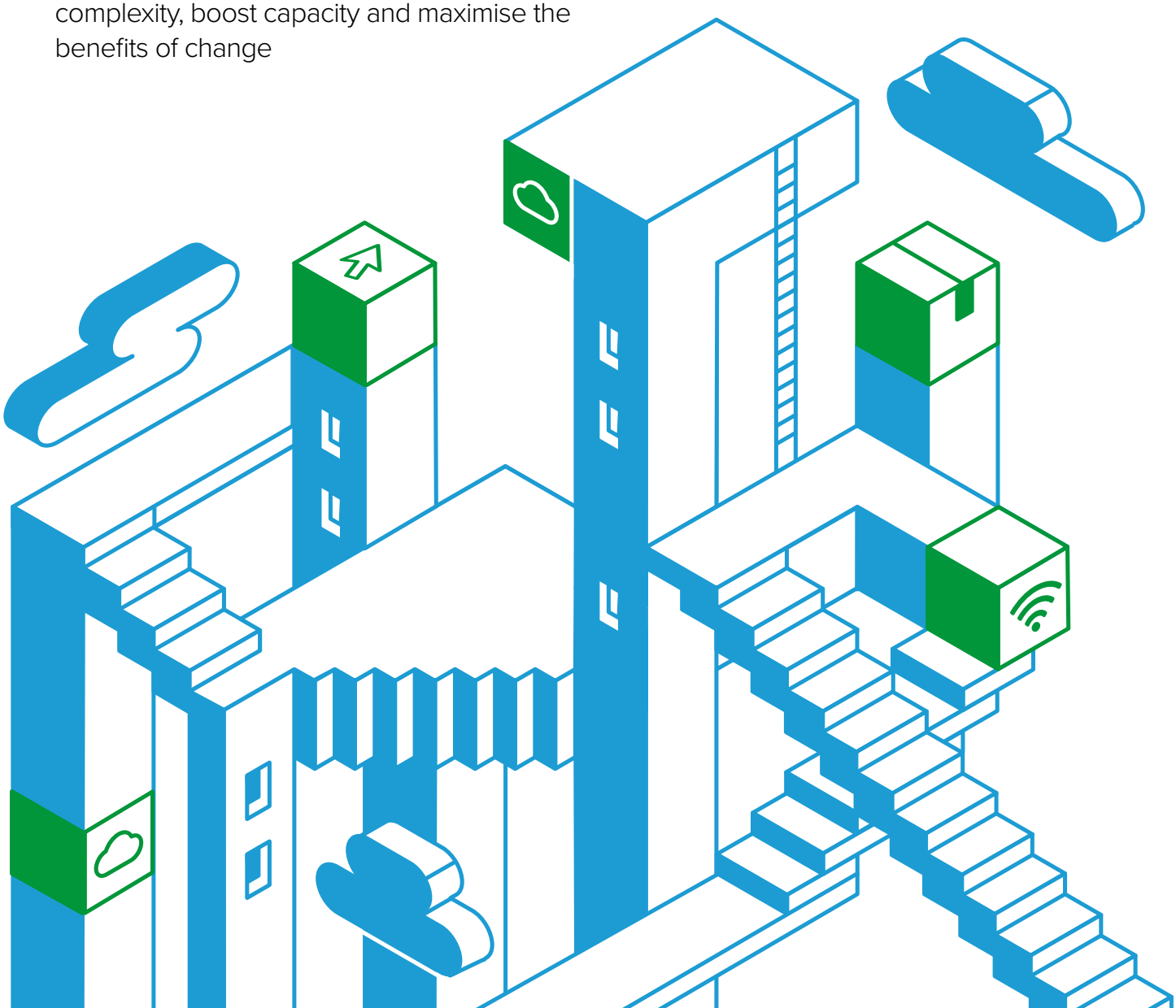
Respondents using Kubernetes in production

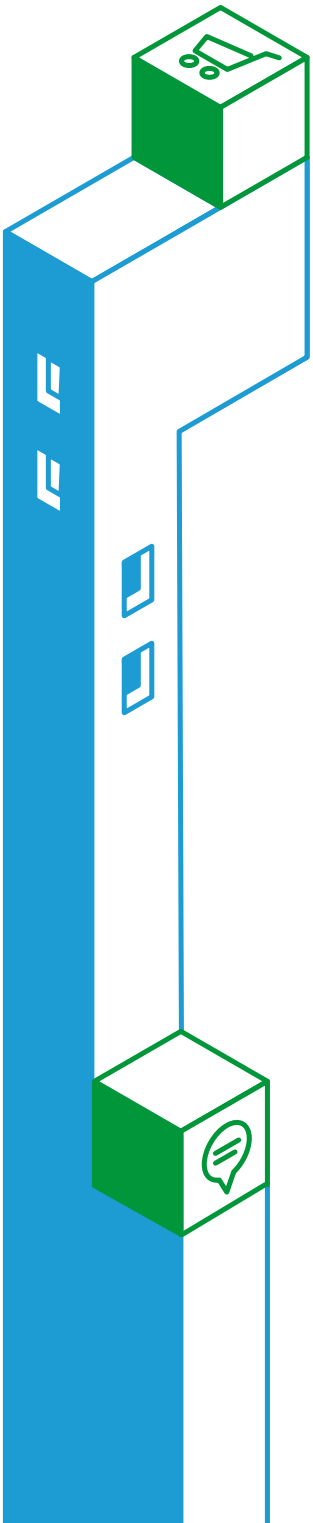


CNCF 2019

Complexity: Common Challenges and How to Solve Them

A trustworthy technology partner can provide valuable third-party support to cut through complexity, boost capacity and maximise the benefits of change





Invariably, bringing in new technologies and methodologies introduces complexities to unpick. As well as wrestling with the finer nuances of these components, there are bound to be fresh challenges that, left unchecked, could lead to the creation of a sprawling architecture that's easy to lose track of. But the other side of that coin is that corralling all of these parts into a well-ordered architecture, and achieving better visibility, is a powerful tool for progress in any organisation's arsenal.

Preparing early can offset risks later, enabling you to get ahead of potential painpoints. First is understanding there are always bound to be new knowledge requirements. The "keep it simple, stupid" (KISS) design principle does not just apply to the US Navy, which popularised the phrase in 1960, and asserts that for systems design everything should be kept as simple as possible. Holding on to this from the beginning of your project will help keep staff conscious of haphazardly adding in new components and assist in avoiding the dangers of runaway bloat or complexity.

"There will always be an antibody in an existing system to resist and say no, I don't want to change," says Dr. Danjue Li, Sr. Director of Incubation Engineering, Product Research and Incubation at Equinix, who leads the engineering effort of incubating new product and service offerings at the data centre provider.

Practically every new piece of technology in a stack will cause some kind of spike in complexity as staff get to grips with it. The questions leaders need to ask are if these will dissipate as users become more familiar and how long these extra demands are expected to last; will they stabilise?

"It's asking whether your spike is going to be justified by the overall benefits that you're getting from adopting that technology," adds Li.

Rob Whiteley, Vice President of Marketing at F5, says: "You have an explosion in complexity just from developers choosing whatever they want," adding that experimental "skunk works" projects can easily snowball into critical applications that have scaled in size and are serving live customers. At that point, businesses realise they wish they'd done things a little differently at the start. But Whiteley compares making changes at that stage to "changing the wheels on the bus while the bus is going down the freeway".

African Bank Technical Architect Fintan Wilson says, from the start, his organisation understood there would be an immediate increase in complexity.

"At the same time, there were other benefits, like being able to scale certain aspects of the application independently," Wilson explains. "That meant something to us and it's something we strive for."

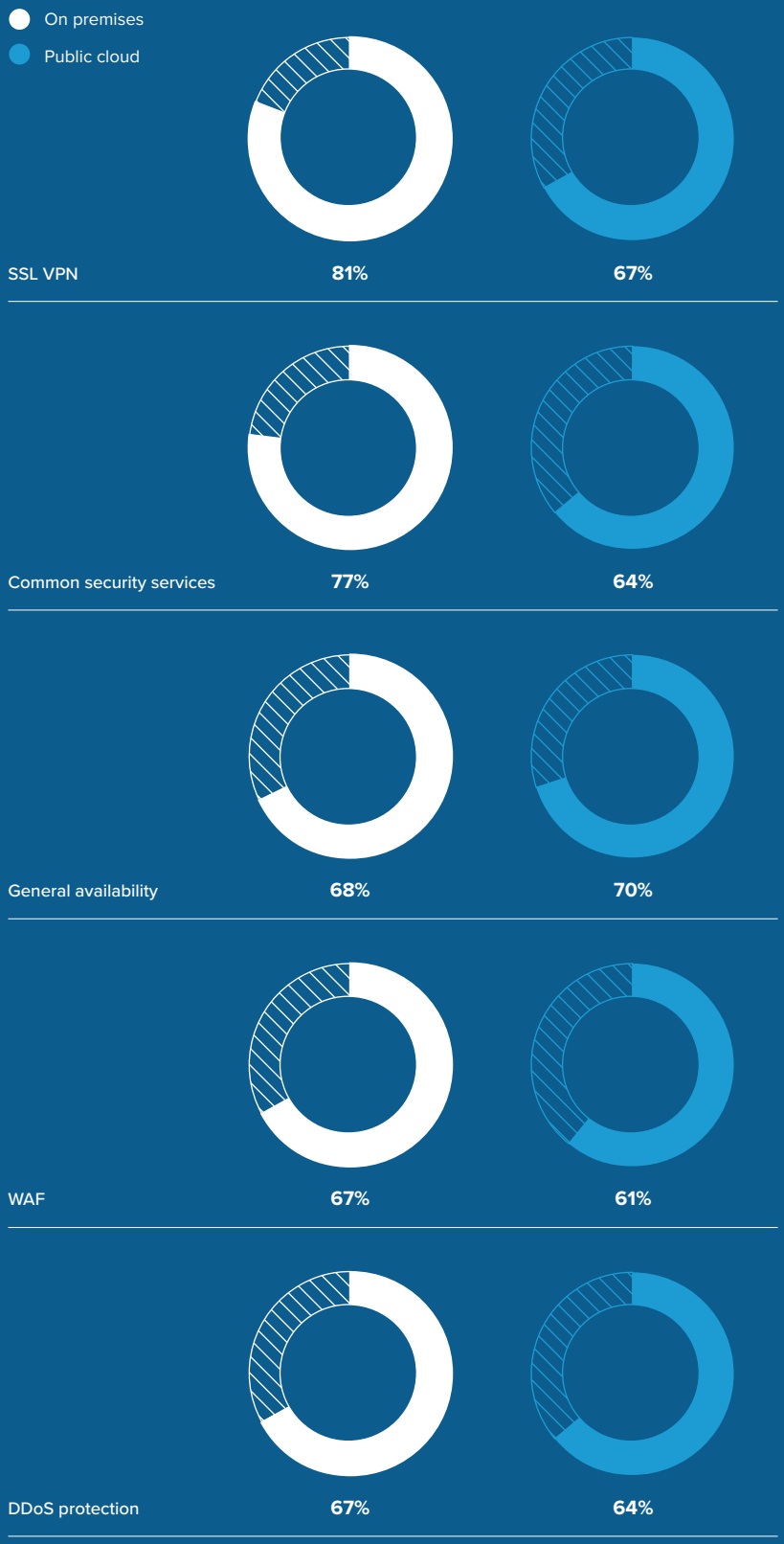
Maximum automation

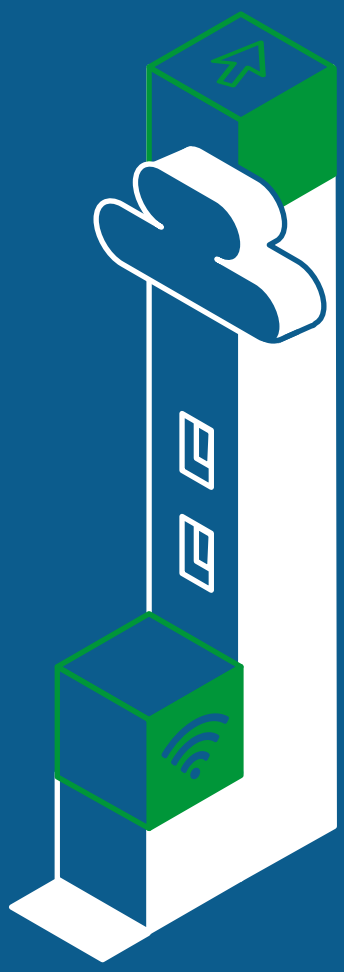
One way the banking group managed the increased workload was by automating as much as it could, starting by writing custom code that would co-ordinate traffic routing. It also enhanced its continuous integration and delivery (CI/CD) pipeline to ensure all builds into Docker images happened automatically.

"As people do make their environments more complex, I think the amount of time they are doing things that were

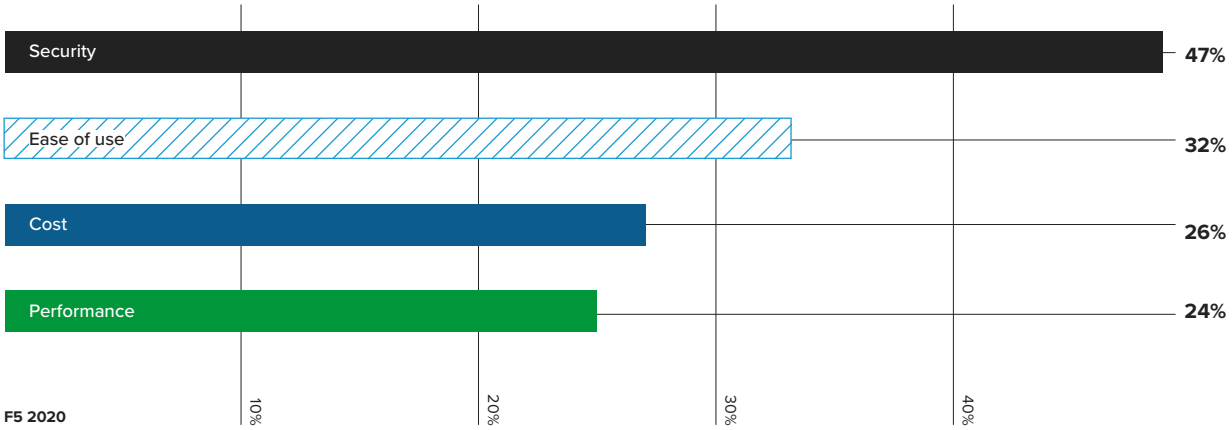
Preparing early can offset risks later, enabling you to get ahead of potential painpoints

Application services currently being deployed





Most important characteristics of an offering when deploying application services



previously mundane has reduced," Wilson adds. "A lot of the focus is now on things that would be more difficult to automate."

Building software, says Li, is merely the first step, and the greater challenge is in how to operate it 24/7 and assert control over its complexity. That means picking a tech stack that can be seamlessly integrated, with enough flexible capacity to scale up when the business demands it.

With all these extra elements talking to one another, there's bound to be hunger for a single domain from which to manage them.

A well-integrated application programming interface (API) gateway is an absolutely essential part of the stack to handle these new environments. Serving as a single point of entry, the gateway protects critical API traffic and harmonises communication between endpoints under a solitary API domain, as well as exposing APIs externally.

Li says when her team first containerised all their workloads, they quickly needed support for web-server caching, API health checks, monitoring, and an API gateway feature to co-ordinate the parts.

As they improved the capabilities of platforms and added more features to APIs, there were extra considerations around versioning, or making changes dynamically to APIs without breaking their functionality.

These needs, as well as requiring a specific way to build multi-tenant APIs where a single API served multiple customers, led Li's team towards NGINX Plus, an enterprise-grade all-in-one software load balancer, API gateway, and content cach built on top of the open source web server and proxy NGINX.

Containerising

But the team also wanted to treat all its infrastructure as code, which meant containerising everything, from databases to applications. This off-the-shelf enterprise option easily integrated with the full stack and, as it could be containerised, fit neatly into the team's CI/CD pipeline for rapid deployment.

As F5's Director for Product Management Liam Crilly puts it: "The problem of being distributed extends itself to observability and traceability" and so monitoring systems are a vital introduction to the stack, to measure the health of all the various parts.

Sometimes, and especially with newer technologies, the expertise may not be on hand internally. Picking a trustworthy technology partner can be key in furnishing a project with third-party support, boosting capacity so your organisation can focus on maximising the benefits you've set out to find, tapping into that extensive pool of specialised knowledge when needed, for example in the instance of any particularly headache-inducing hurdles that crop up.

"Providers will differ on their approaches to technology, but the supporting team from the vendor is among the most important elements," says Li. If you get yourself in a knot, capable support staff with deep technical knowledge on hand as trusted advisers can be a much-needed lifeline.

But in the case of open-source ecosystems, an active community also goes a long way towards demystifying sticking points. You might be surprised to learn just how willing community contributors are towards helping to resolve technical issues; this goes both ways as many developers like to give back, too.

"When there's a large community out there, there's another very powerful source of knowledge that will help us increase the level of understanding and learn best practice," says Li. "Which, in turn, will help us to get the complexity under control."

In short, like everything worth doing, things can and will be challenging at times. But being conscious of architecture design from the start, and tapping into valuable community knowledge as necessary, should prove most problems are solvable.

Learning how to overcome difficulties will also help your organisation achieve a competitive level of technical capability, something that will stand you in good stead for taking advantage of alluring technological prospects and even better business outcomes on the horizon.

The Future is Agility

Laying the foundation to deliver an agile, developer-centric model needn't be fleeting. In fact, starting today, it can go considerable lengths towards future-proofing your organisation for tomorrow

Many of the components for building a microservices-led approach will enable easier integration of software developments to come, especially in high-activity areas in open source.

To understand your preparedness, it's worth clarifying some indicators that you're in healthy shape. One is speed: how quickly it's possible to bring updates live. The best-in-class time for software delivery is around two hours, says F5's Vice President of Marketing Rob Whiteley, although leaders like Google are measuring this time in seconds.

"Most companies should consider themselves successful if they're measuring that in hours," says Whiteley.

That'll be a wild departure from the months many businesses will be accustomed to. If you're hitting these speeds, you'll necessarily have automated much of your pipeline: it'll be interoperable, driven by application programming interfaces, flexible, and built with microservices.

Most companies want to boldly go where five or 10 other companies have already gone

In contrast, manual processes entail enormous amounts of regression testing, ticketing systems, and other laborious tasks, while an automated continuous integration and delivery pipeline runs through as much as it can alone.

To effectively automate these processes means understanding where all the components fit in and how they talk to one another. This visibility is essential to narrow in on problems, but also helps in knowing where there's room for improvement.

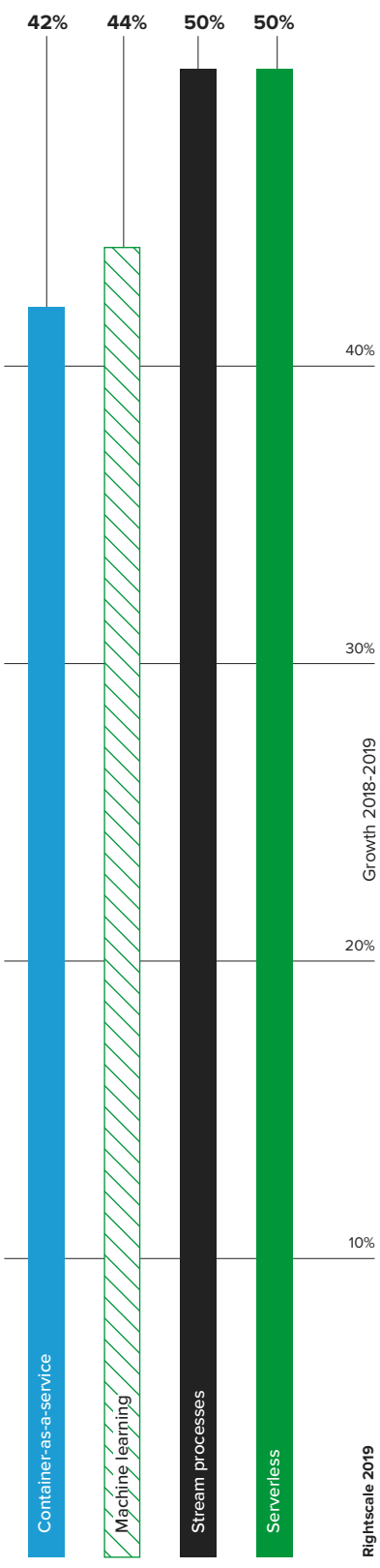
It will then be a case of refinement for crafting an even smoother operation, chipping away at release cycles, and automating more parts.

Compliance challenges

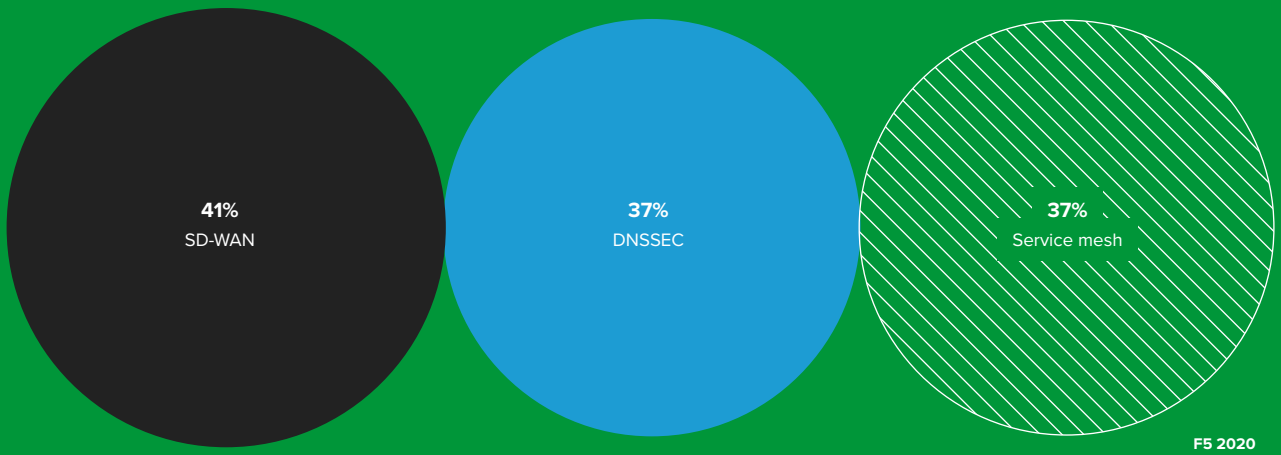
But why might all this matter for tomorrow? Aside from being able to deliver improved core services to customers, achieving better visibility means being well situated for upcoming compliance challenges. Among the first hurdles for businesses preparing for the European Union's General Data Protection Regulation, for instance, was knowing where all the data they owned actually was.

Furthermore, the unmatchable rate of change in open source is leading to

Top-growing extended cloud services



Application services organisations plan to deploy in the next 12 months



F5 2020

developments that, while not yet mature, offer alluring possibilities.

A hot area of development is the service mesh, a low-latency infrastructure layer that ties together high volumes of service-to-service communications. Think of this as east-to-west traffic: services talking to one another on your network, rather than the north-to-south traffic of data flow in and out of your network.

They're designed to help manage distributed microservices, with the service mesh layer touching and controlling every aspect from load balancing, to rate limiting, and authentication with minimal changes to the code of the services themselves. A sidecar proxy attached to the mesh takes the communication between microservices to manage traffic and security.

There's major movement in this space, particularly in the Google-backed project Istio, designed for Kubernetes, although there are others such as Linkerd.

Although some have put these into production, they tend to be for those organisations already at the bleeding edge of computing, businesses running at the level of scale to operate enough microservices to require such a level of management.

"Most companies want to boldly go where five or 10 other companies have already gone," adds F5's Whiteley.

Infrastructure savvy

Developers are more infrastructure savvy than at any other time, but imagine if all they had to do was code. Systems would accommodate developers, automatically allocating resources as they build. Teams wouldn't need to think about storage, servers, or compute at all.

This is the future on the horizon promised by serverless computing. However, while the major public cloud vendors do offer serverless capabilities, they're currently expensive.

Serverless's ability to free up developers' time is without question, but going all-in on a single provider risks coupling you back to infrastructure, which is a regressive step that contradicts with the drive towards decentralising in the first place. Not to mention that unmonitored, the bill could be a nasty surprise.

"I think we need to get to a point where we capture that principle, but in a way where it's a decoupled application service that can run in any cloud," says Whiteley.

In the future, the service mesh and serverless should be complementary, as the former interconnects all the technology of the latter, assisting the code in getting from one function to another.

Throw in advances in artificial intelligence around security, and yet more of the laborious tasks can be given to machines: an intelligent, network-based approach that truly understands dangerous anomalies, resulting in fewer false positives, less time spent fire-fighting alerts, enabling greater focus on designing security into the base of applications.

Together, developments today and tomorrow paint an enticing picture of the future

African Bank had to learn from mistakes to reshape its future, before it could evolve into the digital-first, agile institution it is today. After a lot of hard work, it's now a highly automated, flexible organisation that's well placed to take advantage of future developments and to "turn on a sixpence" should the business need arrive.

Together, developments today and tomorrow paint an enticing picture of the future, where teams are able to work cross-functionally in the most collaborative and efficient possible manner, setting organisations up to focus on what really counts, and leaving the boring stuff to the machines. and leaving the boring stuff to the machines. ■

NGINX®
Part of F5

NGINX started as an open source project in 2004 by Igor Sysoev. NGINX was built to solve the performance and scale for web servers during a time when the Internet was expercienign explosive growth. Igor founded NGINX, Inc in 2011 and in 2013 released NGINX Plus, the first commercial offering that added additional features beyond those in the open source project. Many of these features (advanced load balancing, authentication, caching, high availability) were critical for managing and securing the explosion of traffic found in containers, microservices, and web-scale architectures.

In May 2019, F5 acquired NGINX, adding to its portfolio of market-leading application delivery and security solutions. Now as one of the world's most popular open source projects, NGINX is trusted by more than 450 million sites. It offers a suite of technologies for developing and delivering modern applications. The NGINX Application Platform enables enterprises undergoing digital transformation to modernize legacy, monolithic applications as well as deliver new, microservices-based applications. Companies like Equinix, African Bank, and Dell rely on NGINX to reduce costs, improve resiliency, and speed innovation.

NGINX®
Part of F5