WHITE PAPER

Enable containers in a modern IT: from challenge to good management practices
ABOUT LINKBYNET

Created in 2000, LINKBYNET supports its Clients with reliable and innovative managed services.

The company’s value comes from its ability to govern and manage physical, virtual, cloud and containerized architectures.

Our commitment is to ensure high availability and performance of your applications.

The virtual operations center is worldwide deployed on each continent, ensuring a high level of responsiveness 24 hours a day, 7 days a week.
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Introduction to container technology</td>
</tr>
<tr>
<td></td>
<td>The never-ending quest for agility, responsiveness and... reliability</td>
</tr>
<tr>
<td></td>
<td>The solution to address increasingly complex deployments</td>
</tr>
<tr>
<td></td>
<td>From containers to Docker</td>
</tr>
<tr>
<td></td>
<td>But what exactly is a container?</td>
</tr>
<tr>
<td></td>
<td>Containers and VMs: similarities and differences</td>
</tr>
<tr>
<td></td>
<td>Main advantages of containers</td>
</tr>
<tr>
<td>II</td>
<td>Advantages and limitations of containers</td>
</tr>
<tr>
<td></td>
<td>Real opportunities for companies</td>
</tr>
<tr>
<td></td>
<td>New challenges</td>
</tr>
<tr>
<td>III</td>
<td>What are the best practices for developing high-performance containerized applications?</td>
</tr>
<tr>
<td></td>
<td>Incorporate containers into a global strategy</td>
</tr>
<tr>
<td></td>
<td>Identify real opportunities</td>
</tr>
<tr>
<td></td>
<td>Implement the right development strategy</td>
</tr>
<tr>
<td></td>
<td>Build the operating infrastructure</td>
</tr>
<tr>
<td></td>
<td>Procure the right tools</td>
</tr>
<tr>
<td></td>
<td>Evolve into a software factory</td>
</tr>
<tr>
<td>IV</td>
<td>Managing containers in a modern IS</td>
</tr>
<tr>
<td></td>
<td>Bring the complexity under control</td>
</tr>
<tr>
<td></td>
<td>Avoid the risk and unpredictability of going it alone</td>
</tr>
<tr>
<td></td>
<td>The appeal of a true managed solution</td>
</tr>
<tr>
<td></td>
<td>Focus on the essentials: business</td>
</tr>
<tr>
<td></td>
<td>Security at every stage</td>
</tr>
<tr>
<td></td>
<td>Real unshackled freedom</td>
</tr>
<tr>
<td></td>
<td>The solution that you need</td>
</tr>
</tbody>
</table>
I - Introduction to container technology

According to research and advisory firm Gartner, more than half of global organizations will be running containerized applications in production by 2020 compared to only 20% one year ago. Gartner is not alone in believing that the interest organizations are showing in container technologies has grown exponentially in recent years. Whether start-ups, CAC 40 (French Stock Market index) companies, micro-businesses or key accounts, analysts are unanimous in saying that the use of container technologies in companies is gaining pace.

The never-ending quest for agility, responsiveness and... reliability

It was nearly 20 years ago that the world experienced a major IT revolution as it swept away the trend for so-called «monolithic» architectures in favor of composite applications built from small modules known as services, which can be assembled to suit requirements. The aim is to leverage the legacy infrastructure while improving agility and the speed of response. The key idea is to upgrade the installed base in small steps by only modifying a service, instead of recoding the entire application. The concept is known as SOA (Service-Oriented Architecture) and has continued developing over the years with the emergence of micro services, but the underlying principles had already been established.

This new breed of architecture undeniably offers a wealth of advantages for organizations, but it exacerbates the age-old problem of deployment that takes on exponential proportions in this particular case. What is the point of ramping up development if deployment is held back by slightly different OS versions, inconsistent patches from one system to another, library and framework versions that run with one service and then conflict with another, mixed hardware specifications, variable network configurations and fluctuating security policies? What is already a complex problem quickly turns into a nightmare situation due to the number of services used by an application and their distribution across the information system, all of which injects a high degree of randomness and unpredictability into the software deployment process.

The solution to address increasingly complex deployments

Containers represent the culmination of nearly 20 years’ mature reflection on the issue and offer a simple solution to one of the oldest and most constant problems plaguing IT professionals: how do you ensure the reliable and resilient deployment of an application despite the tremendous number of services, the wide range of computers and

50% of organizations will be running containerized applications by 2020 (source: Gartner)
the differing IT environments? Consequently, it is easy to understand why companies have become so fixated on this technology, especially since several factors have sharpened the need to raise the reliability bar on complex and increasingly in-demand architectures.

The Internet and the millions of users all connected to the same application at the same time have changed the approach to programming, the frequency at which application updates are released and the need for scaling up. The recent explosive growth of the cloud has prompted more and more organizations to pounce upon the advantages inherent in the concepts introduced by microservices by instantiating the same service as many times as required in response to their needs for scalability and accessibility. The different instances are deployed across multiple datacenters (in-house, public and private clouds, etc.), and this hybrid or multi-cloud approach reinforces the need for flexible reliable deployments.

Since 2017, the number of scenarios involving containers with infrastructures comprising smart objects or the IoT has risen. They represent an ideal way of installing and updating applications running on smart objects, especially when those objects have limited power.

**But what exactly is a container?**

**Definition**

A container could be compared to a bubble that isolates the application – or service – from the environment in which it is running. It offers a complete runtime environment, including the application or service and all its dependencies (libraries and binaries), configuration files, parameters and even the versions of the system tools used (navigator, shell, etc.) and the modified system variables (default access paths, registry databases, and so on).

As such, the application or service is no longer affected by any subtle variations in the host system or the underlying infrastructure. Similarly, it can be moved from one environment to another without disrupting its operation. The last and by no means least advantage is that the same environment can support different versions of a containerized application or service without causing any conflicts or incompatibility issues.

**From containers to Docker**

Containers first emerged on the scene over 10 years ago and have gradually come across as a clear solution for addressing all the new challenges relating to elasticity, speed of development, flexible deployment, responsiveness, agility, and obviously reliable deployments.
The rocketing success of Docker

It is hard to talk about containers without mentioning Docker. From a purely technical point of view, Docker is an application virtualization layer that runs and controls the operation of the containers on top of the kernel. In simple terms, it is to containers what hypervisors are to VMs.

Since its emergence in 2014, Docker has truly brought containers into mainstream use by simplifying and standardizing the approach. A container format, instruction sets for creating, deploying and managing containers, an image format for fixing a container as a file that can easily be moved from one environment to another...Docker simplifies the process of creating and handling containers, and even introduces the concept of «repositories», meaning warehouses where developers can store and share their container images. The concept has lost no time in attracting scores of followers. According to Docker Inc., the brains behind the open source solution, over 3.5 million applications have already been containerized using Docker and over 37 billion containers have been downloaded from the company’s repository!

A universal solution

Although Docker immediately comes to mind when talking about containers, it should not be forgotten that there are alternatives, including Rkt by CoreOS (adopted by Red Hat), Containerd, Lxd, CRI-O and Podman. Docker, which was initially designed to run on Linux platforms, has nevertheless gained traction among all market players. Whether different distributions of Linux and Windows, Apple Mac OS X, AWS or Azure cloud platforms, Docker has become the universal solution, and most organizations have implemented the technology.

Containers and VMs: similarities and differences

Virtual machines (VMs) and containers share the same principle, i.e. virtualization, since they both generally ignore the underlying infrastructure. These two technologies often tend to be put in the same basket, but it should be stressed that there are differences. The first is that each VM, unlike a container, includes its own operating system. To run five VMs on a physical server, one hypervisor and five operating systems are required. With containers, only one is required, since the five applications share the operating system’s kernel. In other words, containers are lighter and therefore reduce overhead.

At the same time, VMs are primarily aimed at consolidating infrastructures by replacing several physical servers with a single server that is capable of running images of the previous servers. However, this approach is not without its limitations: To make a VM independent of the infrastructure, the hypervisor emulates a real machine with fixed characteristics. Containers rely directly on the use of kernel system instructions, and are able to use directly the devices recognized by the underlying host. Finally, VMs depend directly on the hypervisor on which they have been created. For example, migrating a VM from VMware to Hyper-V often leads to unpleasant surprises, whereas portability from one system to another is one of the advantages inherent in the containerization technology.
Main advantages of containers

1. **Abstraction**: containers are based on the principle of application virtualization. They are completely independent of the hardware and immune to any variations in the installation, since they include everything that the hosted applications need to run.

2. **Portability**: containers can easily be moved from one machine to another. A considerable advantage in a hybrid cloud environment is that they can be hosted equally on internal servers or a public cloud, and can easily be migrated from one cloud to another.

3. **Densification**: unlike virtual machines (VMs), containers rely directly on the shared operating system kernel, which incorporates all the drivers required to function correctly on the hardware. In addition, containers do not include an operating system and therefore use fewer hardware resources than a VM, which raises performance and simultaneously allows more containers to be incorporated into the same physical machine.

4. **Compactness**: containers only include what the application needs, meaning that they only take up a few kilobytes, unlike a VM.

5. **Speed**: containers can be executed almost instantaneously, whereas a VM needs to be booted up like any machine, and the application cannot be run until the system included in the VM has been fully started. This advantage is often forgotten, but can prove mission-critical in certain situations.

6. **Performance**: by combining the above advantages with an orchestration engine that monitors the microservices performance in real time. A containerized application will generally be perceived as more efficient by the user.

VM-type hardware virtualization attempts to create an abstraction of an entire machine, whereas a container-type application virtualization strives to create an abstraction of the operating system. Since these technologies serve different purposes, they inevitably co-exist, and the fact that an organization uses one does not mean that it will not need the other.
II - Advantages and limitations of containers

Boasting an extraordinary level of modularity, containers are the ideal building blocks when developing a modern IT architecture. They are an invaluable asset for successfully leading a digital transformation strategy for the simple reason that they deliver the flexibility, agility and fast response that all organizations need to gain a foothold in today's digital era.

Real opportunities for companies

Containers are more than just a simple technology. They herald a new way of approaching the IT production chain and unlocking a wealth of benefits:

Easier transition to modern architectures

Microservices are making in-roads into organizations. Spawned by the concepts inherent in the SOA architectural style, microservices translate the determination of IT teams to work more effectively with the business teams by creating programs offering much faster scalability. In practice, they promote an approach to developing software architectures where the applications are formed by standalone services that can be versioned, deployed and managed independently of each other. Microservices may be synonymous with agile development, superior alignment with business needs, greater responsiveness and reduced costs, but their integration often requires considerable effort due to the ever-growing number of microservices.

Containers are ideally suited to supporting microservices and bring a certain level of independence that makes modern architectures easier to deploy and operate. They include directly the programming languages and frameworks that the service needs to run. Therefore, developers no longer need to worry whether interpreter X or library Y will be on the production server: all the necessary dependencies are packaged in the container. To put it simply, the user experience and application behavior will be the same between the machine on which the development is performed and the final production environment.

Clearly not all containerized applications feature microservices, and likewise not all microservices are deployed as containers. But containers offer a much easier way for organizations to migrate to a modern architecture and give greater weight to the relevance of pursuing a microservices strategy.

Streamlining IT budgets

Containers not only use less space and resources than a VM, but they also improve management of the IT infrastructure, especially in the cloud where billing is based on the amount of memory, power and storage used. Containers also have the advantage of only being instantiated when necessary, and their fast start-up times have no or hardly any effect on execution, unlike a VM.

Their extreme portability also makes the switchover from development to operation infinitely more reliable, predictable and reproducible, which shaves considerable time off each stage. The time saved on development and deployment obviously goes hand-in-hand with a significant reduction in costs.
Lastly, as part of the current trend for multi-cloud approaches, combining microservices with containers can pave the way for enhanced strategies depending on the company’s international footprint and the cloud provider’s billing policy. Containers are duplicated as many times as required and are located right next to the user. Whether in China or the United States, users access their applications locally, which drives down network-related costs while improving response times. Similarly, the portability, minimalist design and fast start-up of containers (as opposed to VMs) can be leveraged to instantiate services according to the specific pricing conditions of each cloud provider (time slots, etc.) and thereby lower operating costs.

**Shorter development and deployment cycles**

Finally, in a world where organizations need to be ever faster to respond, containers represent an invaluable asset for putting the IT and business teams on the same page. An application created with containers is easier to manage, since each constituent module is relatively simple and isolated. When it comes to making changes, users merely modify the module without having to rebuild the entire application from scratch. In addition, containers promote a highly iterative approach to developing applications, with each version built from the container of the previous version.

At the same time, containers help streamline the production chain by synergizing collaboration between the development, test and operations teams. Since they are unaffected by the underlying environment, they remain identical throughout the executable delivery chain. In other words, they foster the creation of the shared culture championed by DevOps by simplifying implementation of the «continuous integration» / «continuous delivery» (CI/CD) or «continuous development» concepts associated with the DevOps philosophy. That explains why they have quickly garnered a reputation as one of the best tools for deploying a DevOps approach, despite being completely technology-agnostic.

**Containers, DevOps and microservices: the winning trio**

Containers play an active role in the three cornerstones of the DevOps culture: collaboration (everyone works on the same containers), automation (Docker APIs can be used to automate the different creation, production and configuration tasks) and continuous integration (with its metrics and release management). At the same time, it frees the microservice from all the integration constraints by incorporating the entire environment required for execution. Although these three concepts appeared separately - containers, DevOps and microservices - they are increasingly being combined to deliver applications faster with improved reliability.
New challenges

As with any new technology, the uptake of containers does not come without its share of organizational and technical challenges. Read on for a detailed review...

Team organization and upskilling

Containers and their ecosystem of tools are just some of the new technologies that need to be brought under control by developers, testers and the teams responsible for operations and deployments. Before an organization can adopt containers, its teams require training to gain all the skills necessary to clearly understand all the concept's subtleties and take advantage of the technology's full array of advantages. Training is especially important, since a container-oriented strategy often goes hand-in-hand with a microservices approach, which also demands high-level expertise. Define the right level of granularity, choose the right architecture... in a distributed system that is just as complex as a microservices system, an error in a single component or even a network link can lead to a chain reaction of errors spreading throughout the entire system and hampering or even preventing the application from running.

Finally, containers are the ideal ally for DevOps, since they remove the silos that slow down IT production, meaning that roles can be reallocated and the IT teams reorganized, from development through to operations. Teams need to be restructured to harness a more project-centric strategy (with all the skills incorporated into the project) with the focus on agile development methods and a broader approach including operations.

Bring deployment under control

Containers may simplify the different deployment phases, but they do not eliminate them, especially since they give companies a gateway to even more solutions and opportunities. Containers can be deployed on the internal infrastructure and directly on physical machines, but also on virtual machines, especially for better incorporating them into backup plans, DRPs and existing administration tools, such as System Center and VMware vSphere. Due to their portability, they can also be deployed in the cloud, on leased virtual servers, on container-oriented cloud clusters (such as Kubernetes) managed remotely by the organization and on «serverless container» services like Azure Container Instances (ACI) and AWS Fargate. These last examples are completely server and cluster-independent: developers can place their container in the cloud with a single click without worrying about the machines that will be executing the container.

Irrespective of the deployment strategy, organizations need to learn how to keep a handle on the chosen technologies while acquiring new automation and supervision solutions that are not part of their current installed base. They also need to re-engineer their administration and deployment processes to support containers and their tools.

FaaS: a rival to containers?

The serverless concept is thriving. It is well suited to containers and reduces deployment to just a single click, where the cloud provider’s CaaS infrastructure (Container as a Service) is responsible for provisioning the entire environment. But this approach is currently coming in for competition from a new serverless paradigm known as «Function as a Service» (FaaS), where developers simply write the code for the different functions that are called when the configured events are triggered without having to program all the plumbing that normally gives an application its structure. The FaaS approach allows users to quickly and easily assemble interconnected features with API gateways and event managers without having to deal with the infrastructures, resource management, ramp-up issues and container deployment. As such, FaaS represents an alternative to developing containerized microservices.
Managing containers
In the wake of Docker’s universal adoption, the realization soon dawned that the core market would not flock to the containerization technology itself, but the solutions allowing users to effectively manage containers, just like it quickly became apparent that the core market for VMs did not lie in hypervisor technology, but VM management. When it comes to managing hundreds or even thousands of containers, users need to know what they are doing! Application mirroring, business continuity, load balancing, container lifecycle… a whole ecosystem of solutions has sprung up in recent years to address all the problems associated with managing containers. In this particular field, Kubernetes has practically achieved benchmark status, but the market is still open to a few major competitors, including OpenShift (Red Hat), Docker Swarm, Rancher, Mesosphere DC/OS and Katello. All the main public clouds also offer sophisticated services for hosting, managing and supervising containers, such as Google Kubernetes Engine (GKE), Amazon Elastic Container Service (AWS ECS and its equivalent based on Kubernetes AWS EKS), Azure Container Service (AKS), IBM Cloud Kubernetes Service and OCICEK (Oracle Cloud Infrastructure Container Engine for Kubernetes), etc. In all cases, container management requires additional skills and a level of expertise that cannot be acquired with a snap of the fingers.

Bolstering security
Security is a key concern among companies, and containers admittedly spawn a new host of risks. Although they are completely isolated from each other, all containers are based on the same operating system, or to be more specific, the same kernel, which has its own vulnerabilities and configuration flaws. An attack on the kernel could therefore leave the door wide open and all the hosted containers exposed. That is why some companies prefer deploying containers within VM (in other words, encapsulating containers in VMs) to improve isolation and tighten up security despite losing the inherent performance advantages. To minimize the risks, Gartner advocates the use of an operating system that is as minimalist and hardened as possible.

Furthermore, containers are generally deployed on a management infrastructure, typically a Kubernetes cluster that has its own vulnerabilities (either intrinsic or caused by poor parameter settings).

Should the legacy infrastructure be containerized?
Containers are an integral part of an approach that champions ultra-modular development driven by microservices. They may be capable of being used for any type of development, but their attributes mean that they are vitally important when several teams are working concurrently on independent modules that will ultimately form an application on its own. Although it is fairly easy to design and build a new application with microservices from scratch, it is much harder to transform a legacy application: the application’s architecture needs to be redesigned, which is often a time-consuming and costly process.

Until such time as mature solutions are available for addressing this specific issue with containers, ensuring the security of a container-based infrastructure is a matter for experts.
III - What are the best practices for developing high-performance containerized applications?

Drawing strength from its expertise in managing corporate IT environments and its recently acquired skills in DevOps-related containerization following its takeover of Treeptik in 2017, LINKBYNET has cemented its status as an industry expert with the ability to identify best practices addressing every aspect of designing modern environments:

Incorporate containers into a global strategy

Despite its wealth of promises and actual benefits, containerization should not be thought of as a be-all and end-all solution. In other words, you do not simply create containers for the sake of it. Containerization should be seen as an integral part of a transformation strategy that will enable IT teams to attain the necessary agility to adapt and keep up with the fast pace of change imposed by the business functions, while delivering on the promise to save money on development, maintenance and operation.

Identify real opportunities

Containers may have gained a reputation as the best host for microservices, but their range of uses spreads much further. In some cases, deploying a monolithic application as a container may make perfect sense, since the IT team can leverage some of the flexibility provided by the orchestrators. Even though this approach runs counter to the very concept behind a container strategy, it should not be ruled out without a second thought, since it could turn out to be a judicious move. Nonetheless, organizations should keep in mind that not everything can be containerized.

Too old

Some applications may behave erratically or display incompatibility issues once containerized. This often happens when applications are using near-obsolete versions of Java. The same is true of applications that have been designed for old kernels, but which have not been certified for redeployment on modern kernels. The latest generations of kernels are more suitable to containers.

Excessive system dependencies

Similarly, applications that are overly kernel-dependent use devices (/proc, /sys...) are not suitable candidates for containerization. It is always possible to operate all of this in a container but at the risk of completely losing the «portable» side of the container between two different hosts. It is therefore better to industrialize deployments with other tools.
Excessive data volumes
Lastly, containerizing data-hungry monolithic applications generally goes against recommendations, since the amount of data to be loaded puts the brakes on container start-up times and ending-times (end of business workload). Since they are not very respectful of neighbouring ecosystems, it is often necessary to isolate them, which is possible with the notion of TAINT under Kubernetes. In all these cases, VMs generally have an advantage over containers, i.e. they are not only simple but also faster than having to re-engineer an application to work with containers.

Implement the right development strategy
The container revolution is sweeping the entire software production chain, starting with developments. The «right» approach to successfully leading a project mainly hinges on the organization’s level of maturity in DevOps issues and microservices.

Exercise control over the microservice culture
For companies that have already forged an extensive track record in DevOps and microservice-based development, integrating containers is an obvious choice for ramping up agility, shortening cycles and giving developers full latitude, from writing code to deploying applications. However, creating microservices still represents one of the stumbling blocks. The organization needs real expertise, since the aim is to adopt software architectures providing a clear distinction between the various roles performed by the different services making up the application and thereby avoid a monolithic approach.

Using containers as part of a microservice-based approach to development requires a certain degree of discipline and best practices that tend to be somewhat of a grey area for some organizations. Unlike monitoring, observability is not actually a microservice function but a «character trait». What this specifically means is that developers need to get into the practice of creating «observable» microservices, so that the IT teams can accurately track down a problem in case of a failure or error. Otherwise, applications would become far too unwieldy and costly to maintain on a day-to-day basis, bearing in mind that the services running within the execution cluster (i.e. the orchestrator) are highly distributed.

This particular culture is not naturally ingrained... it needs to be learnt and assimilated by all developers involved in creating modern architectures.

Containers should be ephemeral and stateless, so that they can be shut down, started almost instantaneously and moved with the utmost speed, flexibility and agility. Therefore, the issue of data storage becomes a real challenge, with the implementation of remote storage solutions (NFS, GlusterFS, etc.) to take full advantage of the resilience and agility provided by containers.

Proceed step by step
However, most companies have yet to reach the requisite level of maturity, since their efforts are focused on modernizing the legacy infrastructure. Most of the time, they are planning to overhaul the architecture in a bid to eliminate the silos and break the application down into «roles», which will subsequently take the form of microservices. World-renowned architect Martin Fowler advocates a more pragmatic approach: based on scalability criteria or functionality requiring quick and/or regular updates, he invites companies to start by identifying the business components that would gain from being disconnected and then placed in a container. This approach encourages quick wins and a faster return on investment without preventing the company from gradually harnessing the DevOps philosophy and embracing an advanced microservices culture.
Build the operating infrastructure

As with any form of virtualization, running containers on a physical machine is the culmination of stacking the technology layers to obtain the necessary level of abstraction and the resulting agility. The inherent complexity of this stack calls for varying levels of involvement from the IT teams and requires a wide and diverse range of skills. To wring full advantage from their expected flexibility, containers are implemented using an orchestrator (Kubernetes, Swarm, Mesos, etc.), which manages deployment and execution within a server cluster. The orchestrator forms an extra layer that adds to the overall level of complexity.

Procure the right tools

Keeping the infrastructures under control involves:

- **Regular backups at multiple levels**, including the configuration files of the execution cluster (Kubernetes, OpenShift, Swarm, etc.).
- **Effective image management and monitoring** (such as dockerfiles) to prevent vulnerable or out-of-date images from being deployed in the live environment.
- **Advanced and centralized log management for the applications** containers and orchestrator to ensure that the architecture is performing as planned using monitoring tools to guarantee security.
- **Sophisticated security administration** by making sure that the different layers are free of vulnerabilities, including at the core of the container images.
- **Implementing dashboards** for the IT teams and sometimes even for the business functions, with KPIs tailored to the different profiles.

Getting started in the development area

There is an essential bible to design resilient, scalable and maintainable applications, even if they are not micro-service oriented. Allowing to identify all the issues that IT teams may encounter during hosting, the «Twelve Factor App» guide (https://12factor.net/) offers a clear and concise approach based on a set of best practices.

Evolve into a software factory

To ratchet up their speed of response while keeping a tight rein on their deployments, companies have a vested interest in standardizing all the business processes according to recommended DevOps principles (continuous integration and deployment). Put another way, automating the entire software production chain is the key to operating modern infrastructures with superior efficiency. Fortunately, organizations can evolve in steps and start reaping the benefits of their transformation, even if they have yet to
fully adopt a DevOps approach and incorporate containers into their development processes, for example. In addition to implementing an orchestrator, companies must really concentrate on bringing containers into widespread use and deploying them in the live environment. Achieving such flexibility means migrating to an «infrastructure as code» strategy using such solutions as Ansible and Terraform. In capable hands, these solutions can give the cluster greater elasticity without any need for human involvement.

Raise performance
As with any technology, it takes more than rolling out the right solutions to automatically benefit from effective applications. Containers are no exception to this rule. They require infrastructures to be fine-tuned for peak performance, especially due to the complexity of the different environments, as well as monitoring practices geared towards the different layers (orchestration and underlying environment). However, in the quest for performance with a container approach, efforts need to be invested at the front end of the cycle, such as when auditing, transforming and implementing the application architecture while taking the following three issues into consideration:

• Availability : the design of the containerized application needs to take account of ad hoc and/or seasonal surges in activity.
• Geographic location/accessibility : the application architecture must be designed in such a way that it can be ported to other parts of the world (to bridge the gap between users’ services) while factoring in the often complex issue of synchronization.
• Pure performance : from the outset, the design must consider such key issues as encrypted information and data exchanges, and data storage.

Lead change
Lastly, migrating to a modern architecture is a job best left to experts. It calls for skills, specific processes and real expertise, failing which a pragmatic approach will remain beyond the organization’s reach. Above all, it requires a change in culture: organizations cannot lead their digital transformation strategy unless the IT teams work in tandem with the business functions, which means a radical change in how both sides operate. In too many cases with these projects, companies strike out on their own in the belief that they know the ins and outs of a DevOps philosophy that they have related to implementing a software factory. Even the greatest and most effective software factory is helpless faced with humans who do not have the necessary insights and who stick to their guns because they have not been trained how to cope with the change and have not been given the support and guidance required to follow an approach that is revolutionizing the entire organization.
IV - Managing containers in a modern IS

As we have just seen, the decision to implement and manage a container strategy has a profound effect on the company's development and deployment processes, as well as how its teams are organized. It calls for top-tier skills in creating applications and administrating the new environments, especially in terms of the container orchestrator and the underlying software and hardware infrastructure. Organizations cannot wing it when it comes to best practices. They require on-the-ground experience and an extremely wide skill set.

Bring the complexity under control

One of the defining features of a smooth-running container environment is that it unshackles developers from the constraints of the infrastructure. Containers may conceal the underlying infrastructure, but they do not make it disappear. Many companies toy with the idea of creating their own environment, especially since the process of installing an orchestrator is relatively straightforward. But they tend to misjudge the sheer complexity, time, resources and expertise needed to maintain, upgrade and keep the infrastructure running in peak condition.

Whether considering the operating system hosting the execution cluster or looking at the actual cluster and the associated software, all these components cannot survive without ongoing expertise and administration, such as vulnerabilities, patches and updates. In other words, these skills, processes and best practices tend to slip under the radar when organizations are planning to build their own infrastructure. As a result, examples abound of companies failing to keep pace after just a couple of months and falling behind with their upgrades and updates. Security flaws spread like wildfire. Costs spiral through the roof. SLA targets fall by the wayside. The container promised land becomes a pipe dream...

Avoid the risk and unpredictability of going it alone

In other words and especially with the advent of the cloud, organizations are best advised to avoid venturing down the container road on their own if they wish to reap the full benefits of these new environments. Not only are the technologies new, but they can change in the blink of an eye, meaning that new skills and regular training are essential. The market is awash with solutions, but their maturity does not always live up to their vendors’ claims. Finally, the ready-to-use «Containers as a Service» environments available from cloud providers offer fast-track access to these technologies, but they always require the organization to think long and hard about how to gear its architecture towards its business needs. In addition, the solutions are generally incompatible with each other, which again means that the organization will need specific skills and knowledge for each provider.

Rather than struggling through what seems to be a grueling obstacle course, more and more companies are deciding to save time, free up resources and exploit their creative freedom by seeking out vendors offering end-to-end managed solutions for orchestrating containers. Instead
of wasting time and money on the operational side of the infrastructures - a prerequisite for maintaining an agile environment, but which does not create any intrinsic added value - organizations can ramp up their transformation and focus their attention on the business developments that will bring real added value to the company.

**The appeal of a true managed solution**

LINKBYNET has combined its 20-year facilities management expertise with Treeptik’s specialized skills in container infrastructures to design a managed container solution. Their offering should be seen as a facilitator allowing companies to adopt, implement and operate containers with greater ease. The only one of its kind in the market, this solution is built around three focus areas:

- **Technology expertise**: best-in-class skills in containers and consequently the cloud solutions and tools used by the company’s different BUs.

- **Training and consulting**: pursuing a container strategy naturally ushers in a change in organizational culture, which is implemented by the DevOps philosophy and the principles of continuous delivery. The aim is to improve responsiveness and agility. The teams at Treeptik/LINKBYNET can draw on their extensive knowledge of all these concepts to support and guide companies with their transformation plans by providing insights and putting the new development philosophies into action. They work closely alongside the company’s teams, whether developing their skills, successfully leading a DevOps transformation project or actively driving project
development and execution. LINKBYNET is determined to achieve its objective of changing mindsets and promoting a centralized governance approach to leading change and operations.

**Incorporation of specific needs:** the bespoke LINKBYNET solution factors in and adapts to the specific needs of each company. If necessary, the solution includes environments creation and obviously provides end-to-end management of the execution cluster, while monitoring container performance with an advanced leading-edge incident management system. Not only is the solution’s scalable packaging designed to evaporate the company’s constraints, but also avoid imposing or locking the organization into predetermined technologies.

**SLA commitments:** the fully managed LINKBYNET solution is automatically bundled with SLAs for ensuring the performance and availability of the containerized infrastructures. The SLAs ensure consistent performance regardless of the technologies used and provide full peace of mind for both information system divisions and the business departments using the infrastructures.

**Focus on the essentials: business**

Engineered to provide organizations with all the agility that they need to spearhead their digital transformation, the LINKBYNET solution addresses all the requirements relating to correct container execution to deliver a high level of security, resilience, reliability and quality of service in accordance with organizations’ requirements.

With this aim in mind, it incorporates the elements and operations that the IT teams often overlook or neglect, such as backing up the cluster configurations or managing the images. It also includes maintenance of the infrastructure’s building blocks, whether managing vulnerabilities, upgrading to new versions of the underlying middleware and layers, or updating the operating system hosting the orchestrator. The entire infrastructure is clearly kept under close surveillance with LINKBYNET-enhanced monitoring tools to produce advanced metrics for understanding how each microservice is behaving. When these metrics are combined in dashboards, they offer an undeniable advantage for quickly pinpointing any bottlenecks and fine-tuning the orchestrator’s parameter settings, which in turn raises the application performance bar.

In other words, LINKBYNET provides the necessary maintenance, availability, updates and operations on all the layers stacked inside the container environment.

**Security at every stage**

Lastly, various studies have shown that security represents one of the main obstacles preventing containers from entering mainstream use in companies. LINKBYNET has drawn on the expertise developed in recent years to implement a number of innovative processes for improving container security.

For example, its solution features specialized tools for defining true comprehensive security policies, including control of the images sent by developers in the user company. Images are regularly scanned to detect any known flaws and then define security policies, so that any images failing to fulfil specific constraints or exceeding given vulnerability thresholds cannot be deployed by the orchestrator. At the same time, the technical operations teams at LINKBYNET check that the Linux system underlying any container orchestration infrastructure includes robust security mechanisms and that the orchestrator is suitably protected and configured to close any loopholes.

---

**The new face of managed services**

Empowered by 18 years’ experience in delivering facilities management services for hardware and applications, LINKBYNET has carved a reputation as a trusted partner capable of modernizing its approaches and keeping pace with the latest technological and organizational changes. Managed services is primarily the art of providing support and guidance. Bolstered by its wide array of expertise and the new technology paradigms, LINKBYNET helps companies digitally transform their architectures, organizational structure and applications.
Furthermore, LINKBYNET has built a real identity management system (similar to Red Hat SSO) on top of the orchestrators to tighten access to the production infrastructures. Finally, proactive automated systems flag any incidents or poor practices to improve the operation of the environment while enhancing security.

**Real unshackled freedom**

In theory, containers are portable, but in the real world, depending on which technologies and cloud providers are used, their portability may be hampered by specific security constraints or varying implementations from one cloud to another and from one orchestrator to another. Containers would then lose one of their main assets, since portability is not only synonymous with agility and responsiveness, but it also prevents organizations from becoming dependent on the vendor. Nonetheless, best practices are available to sidestep this trap. They require a certain level of expertise and the right tools. That is exactly what LINKBYNET gives its customers by shielding them from the different implementations and incompatibilities invariably introduced by the «Containers as a Service» solutions offered by the main cloud providers. Thanks to its insights into the solutions developed by the different cloud providers, LINKBYNET can guarantee its customers a gateway to true portability.

**The solution that you need**
From 2020, 50% of companies will have containerized applications in production.
Gartner datas 2018

And you, where are you at ?

linkbynet.com
Info-linkbynet@linkbynet.com