SPARKFABRIK

SPARKFABRIK **WHITE PAPERS**

GUIDA ALL'ADOZIONE DI KUBERNETES

I PASSI DA COMPIERE, GLI ERRORI DA EVITARE, I MIGLIORI TOOL DA USARE

Kubernetes non ha bisogno di presentazioni.

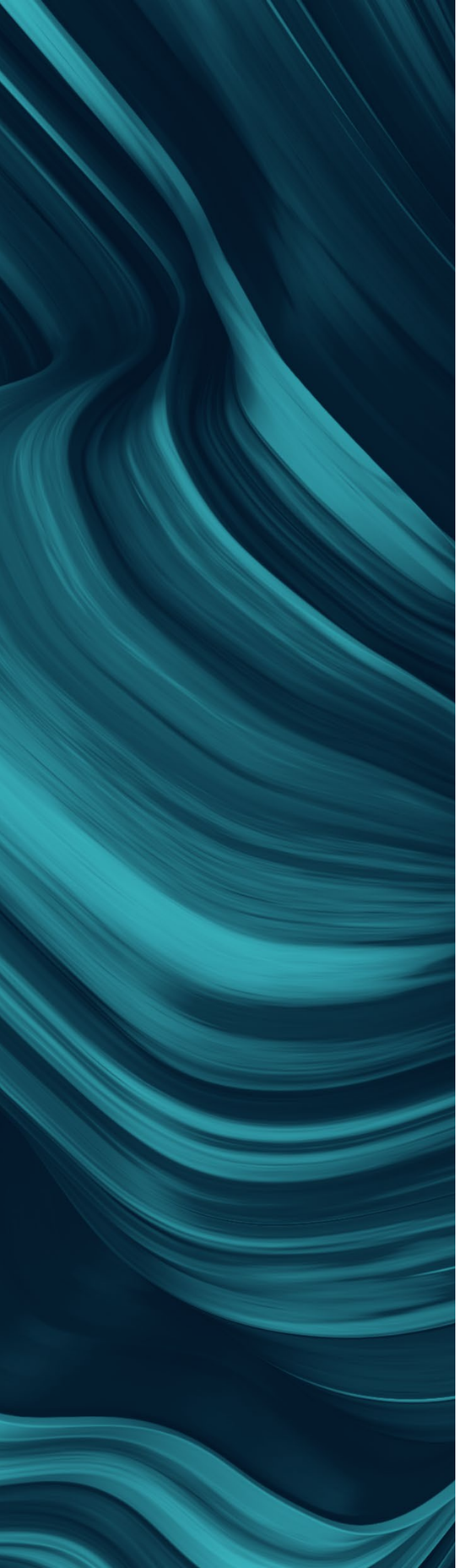
Standard *de facto* per l'orchestrazione dei container, Kubernetes come lo conosciamo **nasce nel 2014 da un progetto interno di Google**, Borg. La decisione di renderlo open source è stata ripagata dalla collaborazione di numerosi software engineer, che hanno contribuito a far crescere il progetto e a stimolarne l'adozione.

Ad oggi, K8s ha sempre lo stesso obiettivo delle origini: **rendere più efficiente l'utilizzo delle risorse hardware,** evitando le spese inutili e l'allocazione statica di risorse che nella maggior parte dei casi rimangono inutilizzate. Risparmio, efficienza e disponibilità sono, in breve, i benefici cruciali derivanti dall'utilizzo di Kubernetes.

Se da un lato i vantaggi di questa tecnologia sono chiari, dall'altro **la sua adozione solleva numerosi dubbi**. Lo stack IT e il team sono pronti per tale cambiamento? Quale roadmap di implementazione è meglio seguire? Quali sono gli errori più comuni che le aziende commettono?

In questo White Paper pratico dedicato ai responsabili Digital e IT risponderemo a queste e altre domande, delineando le best practice per cogliere tutti i vantaggi di Kubernetes senza incorrere in ostacoli che compromettano il risultato.

INDICE



I PRE-REQUISITI

LA TUA AZIENDA È PRONTA PER KUBERNETES?

Per quanto ne valga la pena, introdurre Kubernetes non è esattamente semplice come bere un bicchier d'acqua. Prima di lanciarsi in questa impresa, **è bene valutare la propria situazione di partenza**. Di seguito analizziamo in particolare le skill, le tecnologie e i mindset che dovresti fare tuoi se vuoi avere successo con l'implementazione di K8s.

CONOSCENZE

Per quanto apparentemente banale, è meglio sottolinearlo: è fondamentale **avere conoscenza della logica dei container**. A tal proposito, **Docker** e **Docker Compose** sono due tool ottimi per iniziare perché condividono alcune funzionalità base con Kubernetes. Docker è lo standard *de facto* per gestire ed eseguire container, mentre Compose è un orchestratore molto semplificato rispetto a Kubernetes e adatto all'ambiente locale. Comprendere e sapere utilizzare questi due tool significa mettere il primo tassello per poi arrivare a gestire la complessità di Kubernetes.

In secondo luogo è necessario **conoscere la sintassi YAML**, usata per scrivere i manifest che dichiarano lo stato desiderato del sistema su Kubernetes. Saper comprendere il contenuto dei manifest, in un sistema basato su logica dichiarativa come Kubernetes, non è un optional.

Inoltre è molto utile (ma non indispensabile) avere una minima **conoscenza del linguaggio GO**. Infatti K8s è interamente scritto in GO, che è il linguaggio più spesso adottato anche dagli sviluppatori di componenti terzi. In alcuni casi sapersi districare nei sorgenti delle applicazioni che vengono installate nel cluster potrebbe essere la strada più rapida per analizzare comportamenti inaspettati.

REQUISITI APPLICATIVI

Per poter anche solo pensare di adottare Kubernetes, le tue applicazioni devono essere rese portabili in un ambiente containerizzato, se già non lo sono. Per farlo l'ideale è seguire i principi della **Twelve Factor Methodology**, ovvero le dodici linee guida per sviluppare applicazioni Cloud Native. Questa esigenza porta con sé un importante dilemma: meglio modernizzare l'applicazione o ricostruirla da zero? Per gli ambienti legacy vale la pena valutare seriamente le due ipotesi in virtù di tempi, costi e risultati.

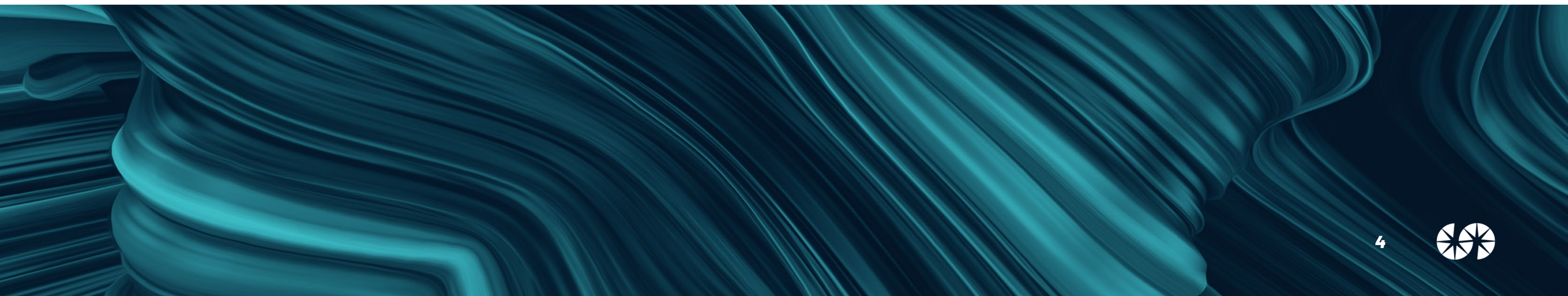
Ma non basta. Infatti **l'applicazione deve anche essere capace di scalare orizzontalmente** per poter usufruire dell'**Horizontal Pod Autoscaler** di Kubernetes. Si tratta di una funzionalità indispensabile per gestire automaticamente e rapidamente i picchi di carico: quando il traffico aumenta, l'HPA lancia automaticamente altre n istanze della stessa applicazione su cui indirizzare il traffico per assicurare la continuità di servizio.

MINDSET

Lavorare con Kubernetes richiede anche di cambiare il proprio mindset, in particolare:

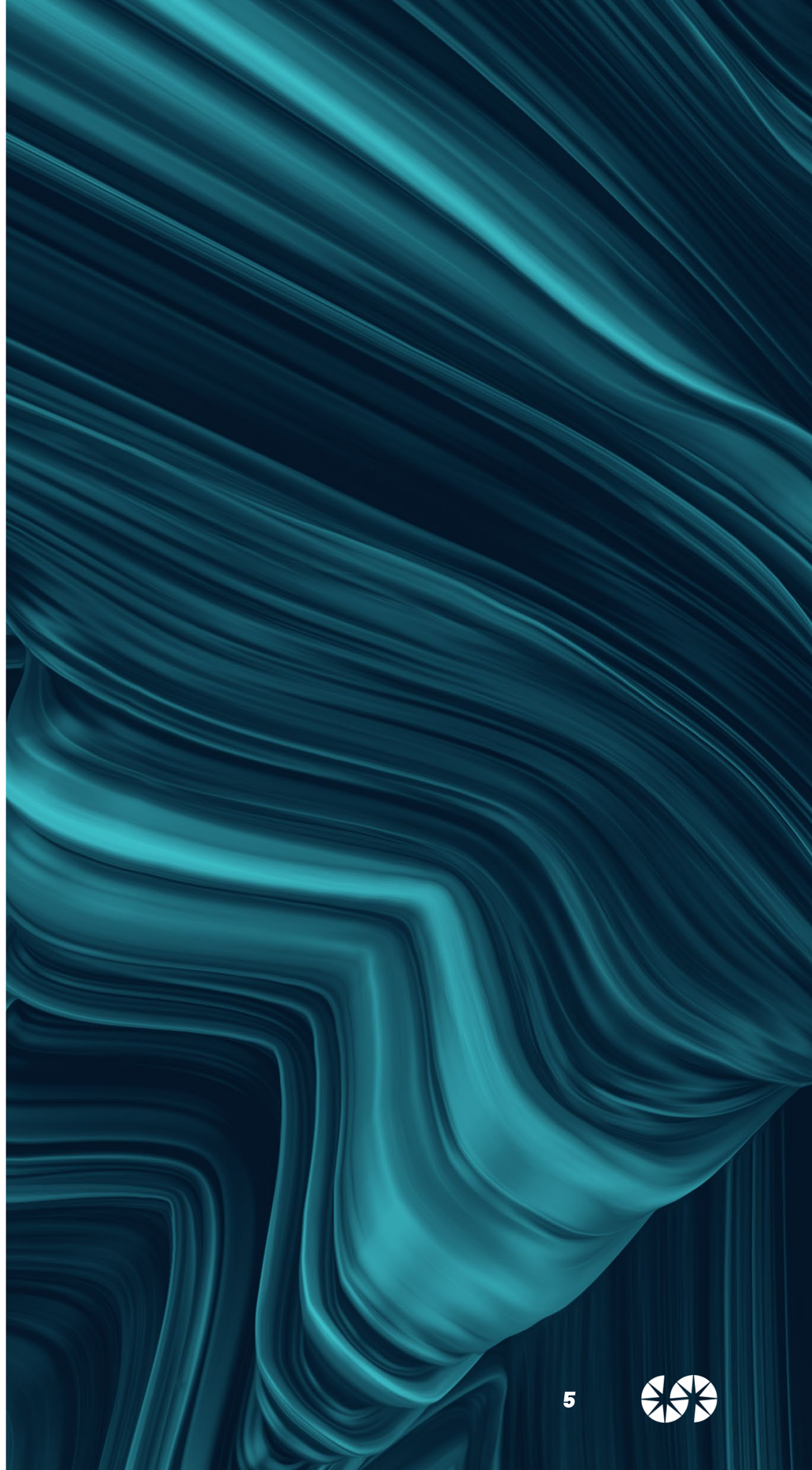
- usare la sintassi dichiarativa
- abbracciare il paradigma DevOps
- applicare la *security by design*

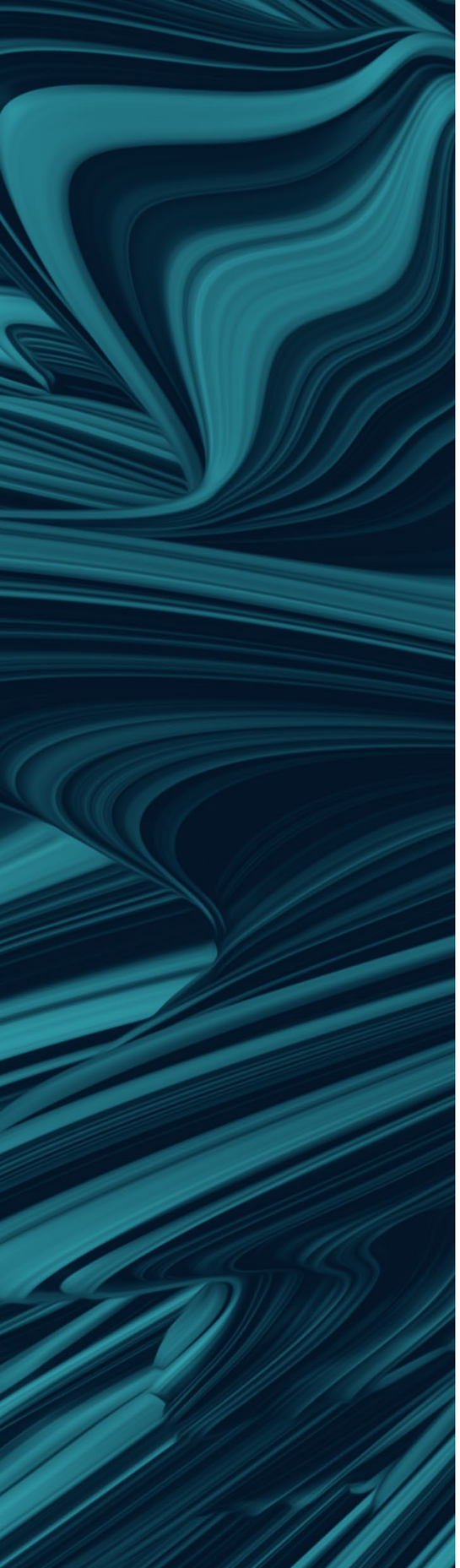
Il primo punto prevede il passaggio da un approccio imperativo, in cui si indicano al sistema le istruzioni necessarie per raggiungere lo stato desiderato, a un **approccio dichiarativo**, in cui invece si indica direttamente al sistema lo stato desiderato. Sarà poi il sistema stesso a mettere in atto tutte le azioni necessarie per raggiungerlo, verificando costantemente lo stato attuale per portarlo a quello desiderato. Questo processo è chiamato **reconciliation loop** e su Kubernetes è operato da un controller che può essere built-in o esterno.



Il secondo mindset che è consigliabile adottare è quello DevOps. Anche se non prettamente indispensabile per usare Kubernetes, implementare il **paradigma DevOps** e di conseguenza l**a CI/CD e l'automation** consente di trarre il massimo beneficio da K8s. Senza un ecosistema completo, Kubernetes rimane un mero esecutore che presuppone comunque molto lavoro manuale: diventa quindi difficile, se non impossibile, sfruttare al massimo le potenzialità offerte dalla tecnologia.

Infine è di fondamentale importanza entrare nell'ottica della ***security by design***, principio chiave del DevSecOps. Si tratta di un aspetto sempre più discusso ma paradossalmente raramente davvero messo in pratica, complici la complessità aggiunta nelle applicazioni moderne e l'esigenza di tempi di rilascio più brevi. Occorre quindi tenersi aggiornati sulle migliori best practice e i pattern consolidati per lo sviluppo di applicazioni Cloud Native, oltre a **focalizzarsi fortemente sull'osservabilità**, onde evitare un utilizzo eccessivo di risorse causato da bug.





ROADMAP E BEST PRACTICE D'ADOZIONE

Veniamo ad alcuni consigli per inserire Kubernetes nel proprio ecosistema IT e nei propri processi.

Innanzitutto, prima di adottare lo strumento è il caso di analizzare il proprio stack tecnologico per **pianificare al meglio la transizione**. In questa fase sono molti gli aspetti da considerare con cura. Occorre valutare le dipendenze applicative o di configurazione, i requirement e i tool di sicurezza, nonché il modello di sviluppo e di deployment.

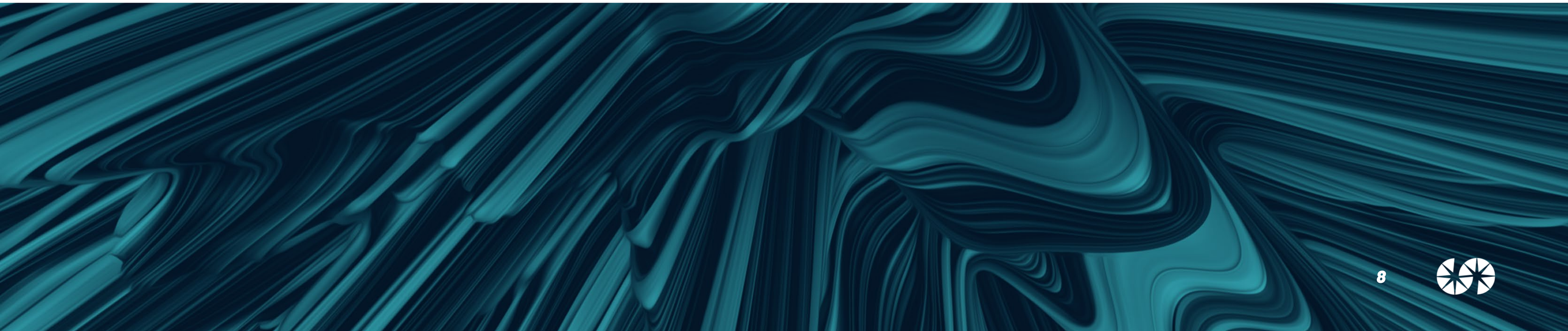
Parallelamente, come descritto nei pre-requisiti, è d'obbligo prendere familiarità con tutto l'ecosistema per poter avere una chiara idea del suo funzionamento. Sicuramente della **formazione di base**, proposta anche sul **sito di kubernetes**, risulta fondamentale per iniziare il processo di apprendimento. Con l'andare del tempo, poi, **mantenere un livello alto di aggiornamento** permette di seguire le best practice e i pattern più all'avanguardia (che potrebbero anche risolvere o mitigare problemi di sicurezza).

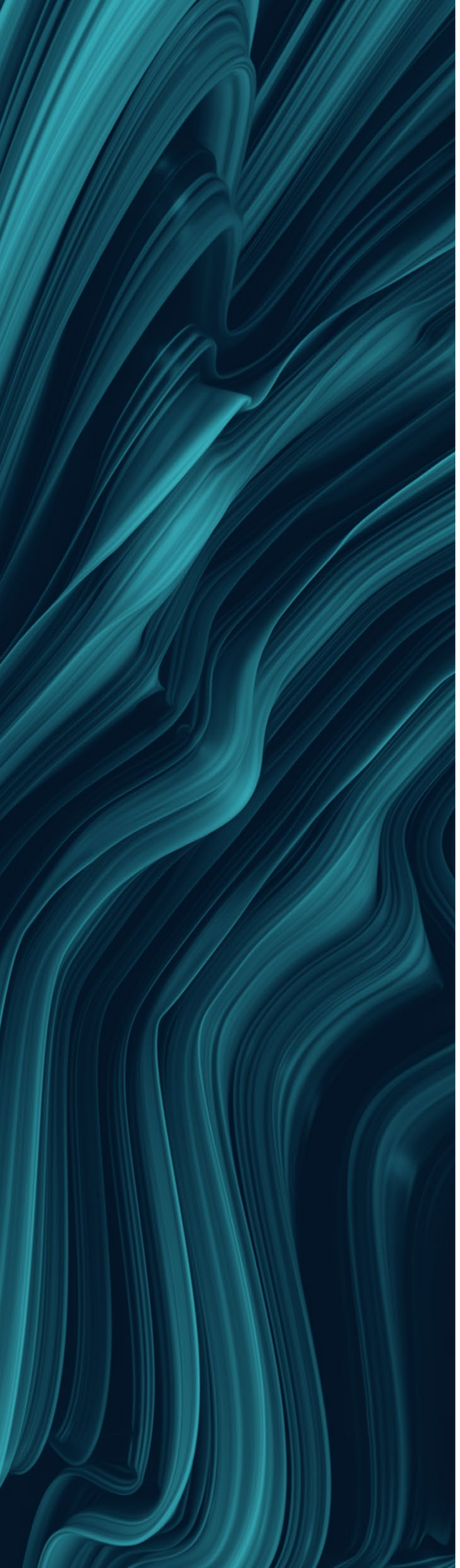
Inutile negarlo, Kubernetes porta con sé una certa complessità. Per favorirne l'adozione è quindi consigliabile concentrare gli sforzi sulla **comprensione del funzionamento della piattaforma** piuttosto che investire le proprie energie nei dettagli tecnici di installazione, messa in sicurezza, aggiornamento, ecc. Il processo per ottenere e mantenere un'installazione di K8s *production ready* può essere condotto da un fornitore cloud. Affidarsi a **servizi gestiti dei cloud vendor** permette di sfruttare immediatamente i benefici che la piattaforma fornisce out of the box; successivamente si potranno mettere in cantiere eventuali processi di personalizzazione per integrare efficacemente i propri flussi all'interno di Kubernetes.

Una domanda comune riguarda la quantità di applicazioni da coinvolgere nel processo. L'adozione progressiva è la strada più semplice da seguire: **sperimentare una nuova tecnologia scegliendo un team o un prodotto circoscritto** permette di fare considerazioni più realistiche e deterministiche. Adottare K8s seguendo questo pattern permette di testare la piattaforma esponendoci il meno possibile al rischio, sperimentare le sue funzionalità e testare i comportamenti in modo più circoscritto,validando le eventuali retroazioni con più sicurezza.

Le persone formate durante questa fase, dato il know-how acquisito, potranno a loro volta diventare i **responsabili della migrazione di altri team/prodotti**. In caso di necessità è possibile anche avvalersi di esperti esterni, che agiranno in un ambiente segregato rispetto all'intero flusso produttivo aziendale.

Infine, come già sottolineato tra i pre-requisiti, è necessario
tenere presente che iniziare a usare Kubernetes non è sufficiente. K8s è la piattaforma su cui effettuare i deploy applicativi, ma i benefici più ampi si ottengono quando **tutto il processo è armonico,** partendo dai repository, passando per le pipeline di CI/CD, per arrivare allo stage di deployment su Kubernetes.

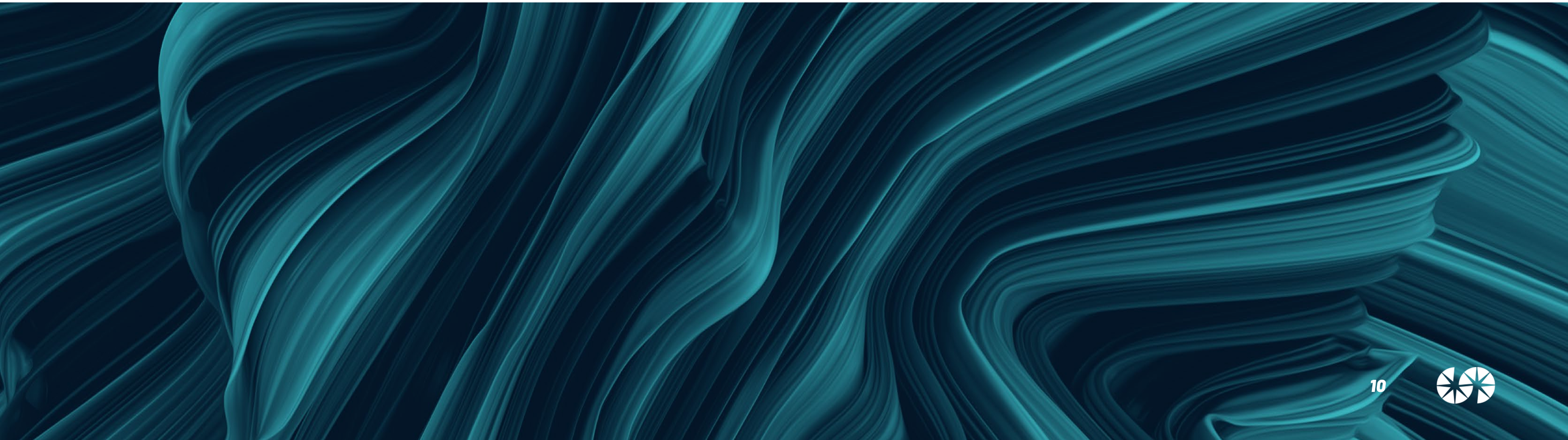




ERRORI DA EVITARE

1. CREDERE CHE KUBERNETES RISOLVA OGNI PROBLEMA

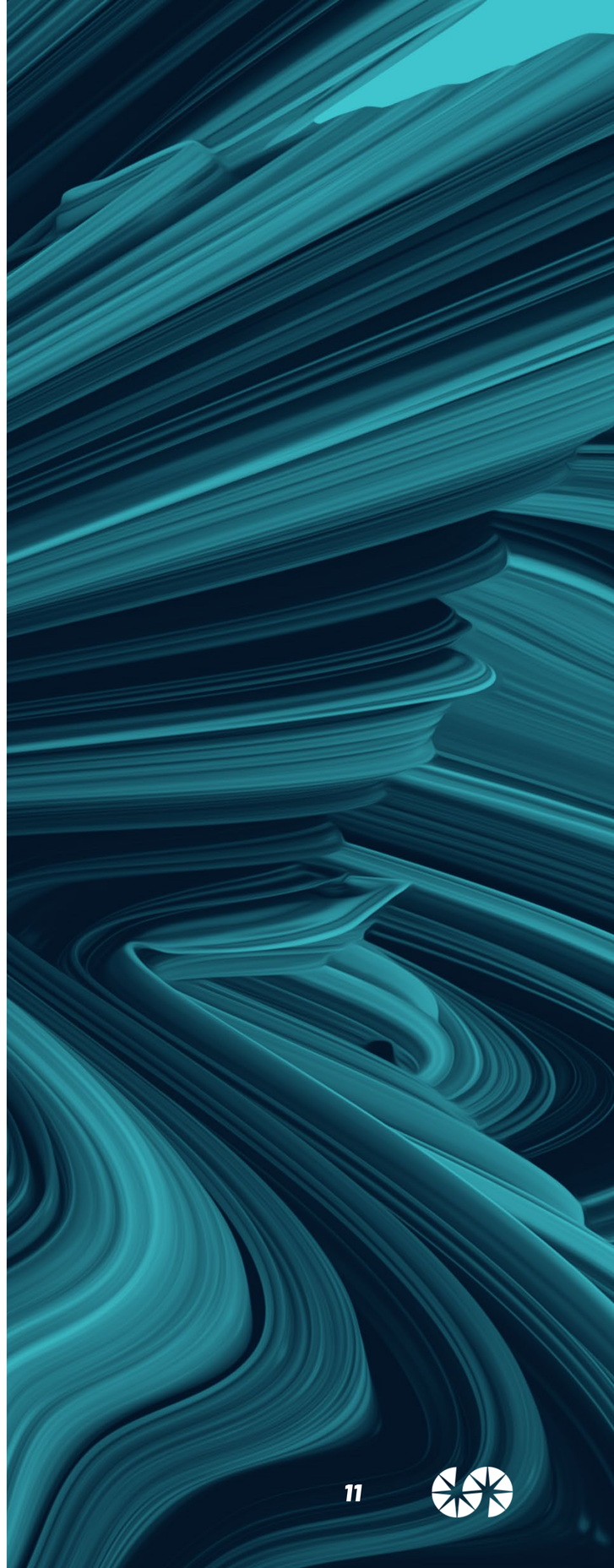
Attenzione a non pensare che K8s sia la medicina a qualunque male. Data la vastità delle feature offerte e la loro complessità, **l'architettura risultante non è facile da dominare**; proprio per questo sono stati creati differenti percorsi formativi per diversi livelli o aspetti del sistema con relative certificazioni. Ecco perché bisognerebbe sempre operare le proprie scelte con piena consapevolezza di quelle che saranno le conseguenze, tanto a livello di vantaggi quanto a livello di sfide.

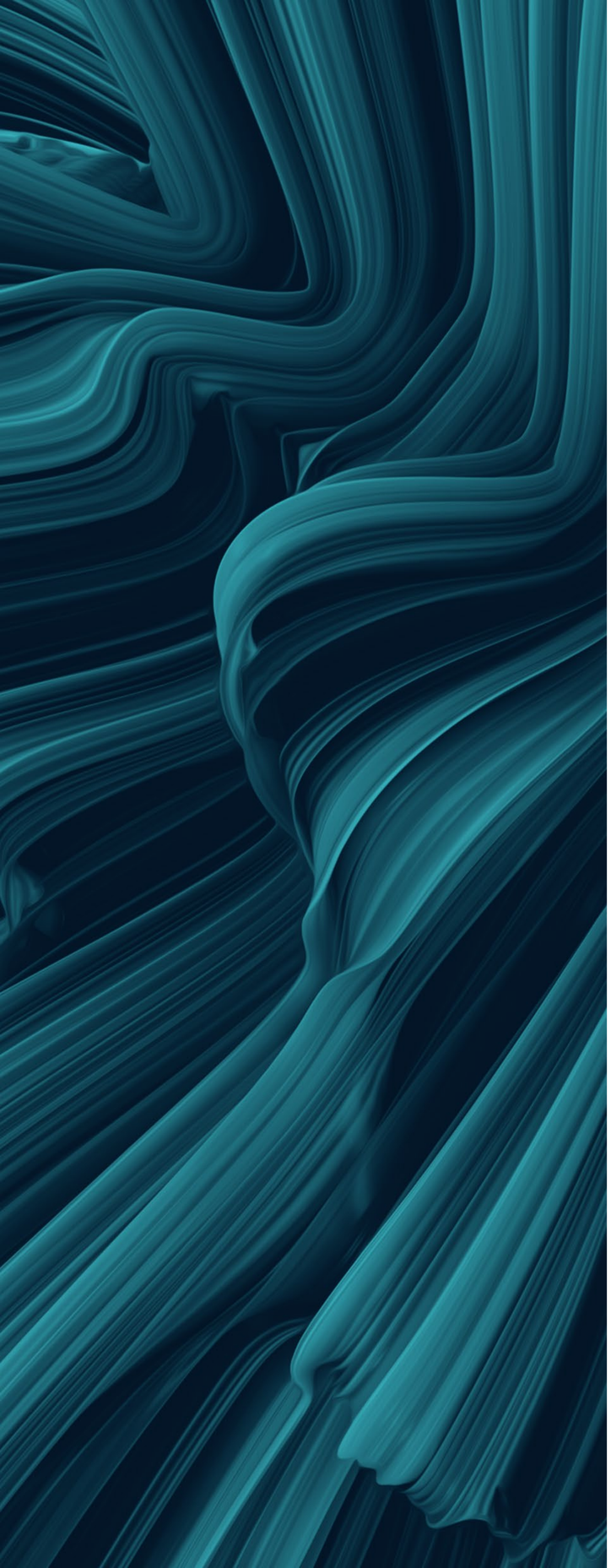


2. ADOTTARE K8S PERCHÉ "TUTTI USANO K8S"

Scegliere una soluzione tecnologica in base alle mode del momento non è mai una buona idea. Nel caso di Kubernetes, talvolta i servizi di cui si ha bisogno sono già offerti come **soluzione gestita chiavi in mano dai cloud vendor**: in queste situazioni non ha senso aggiungere complessità, strutture e costi inutili.

In definitiva, occorre valutare attentamente cosa scegliere in base alle proprie esigenze. Per alcune aziende, ad esempio, la soluzione più adeguata potrebbe essere **un'architettura serverless**, che ha una curva di apprendimento più dolce e possibilità di rettifica in corso d'opera più semplice.





3. SOTTOVALUTARE LA COMPLESSITÀ DI KUBERNETES

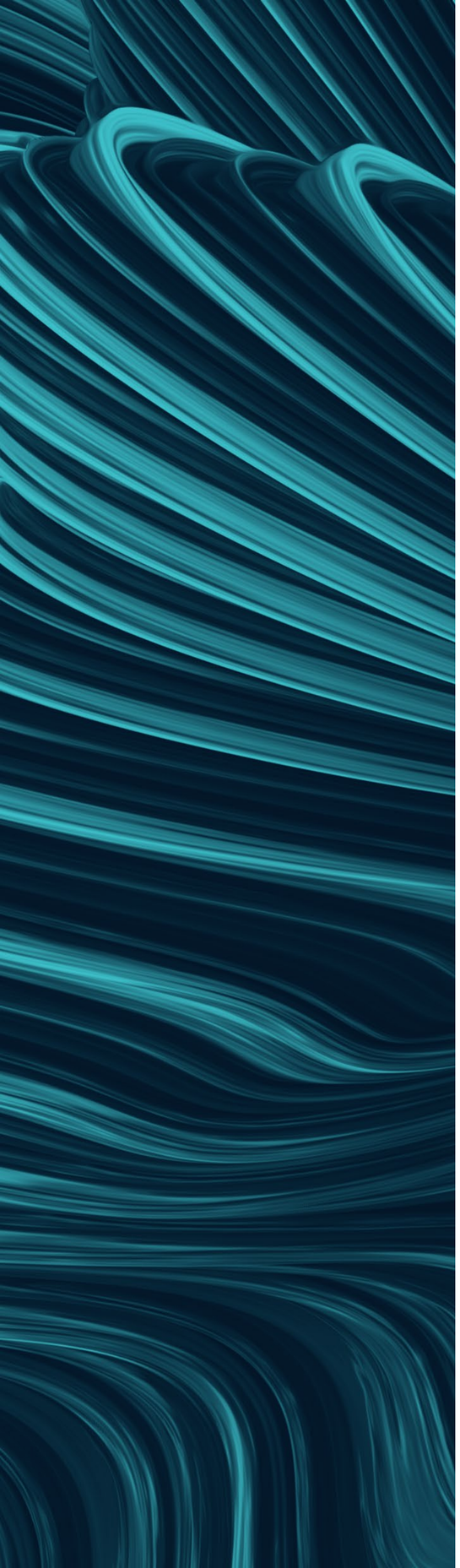
Può capitare. Si testano sistemi K8s locali come Minikube o Kind e tutto sembra molto semplice. Ed ecco il pensiero fallace: "potrei semplicemente **riconvertire l'hardware fisico che già possiedo ed installarci sopra Kubernetes"**. Per quanto possa sembrare una buona strada, questa opzione risulta spesso fallimentare a causa della complessità di mantenimento e gestione del sistema.

A meno di non essere vincolati da particolari policy aziendali che impongono l'utilizzo dell'hardware in-house, quindi, la soluzione migliore per partire è **affidarsi ai servizi di un cloud vendor**. Così facendo sarà possibile valutare la validità della soluzione e capire in che direzione procedere dopo un test sul campo: la direzione giusta, a questo punto, potrebbe anche essere l'installazione di Kubernetes sui server di proprietà. In ogni caso sarà una decisione presa in modo più consapevole.

4. NON ANALIZZARE I COSTI PREVENTIVAMENTE

Kubernetes comporta dei costi tanto in fase di implementazione quanto nel periodo successivo, in cui bisogna considerare i **costi di manutenzione** e il tempo che il team Operations dovrà dedicarvi.

In una fase preliminare è bene mettere in conto questa spesa, insieme ad eventuali **costi di modernizzazione** in caso si tratti di un'applicazione legacy. Avere una giusta panoramica dei costi permette di sapere a cosa si andrà incontro e a valutare se la soluzione faccia al caso proprio.

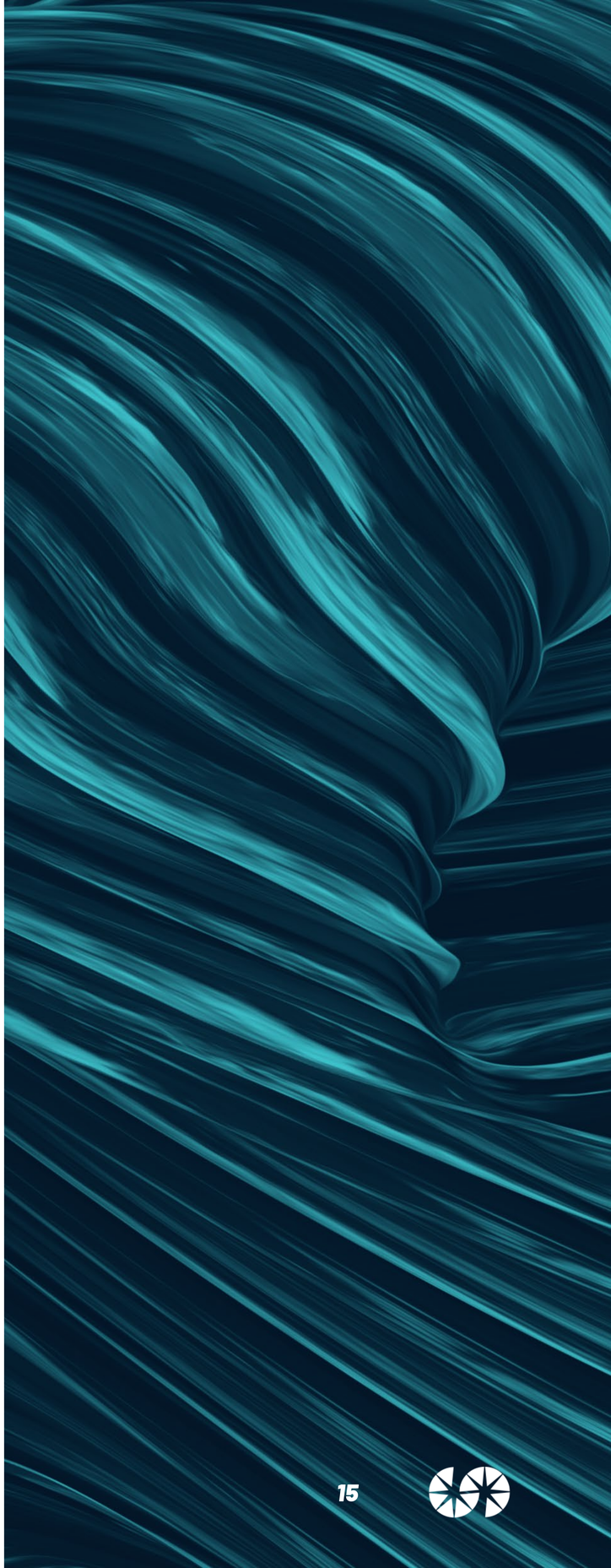


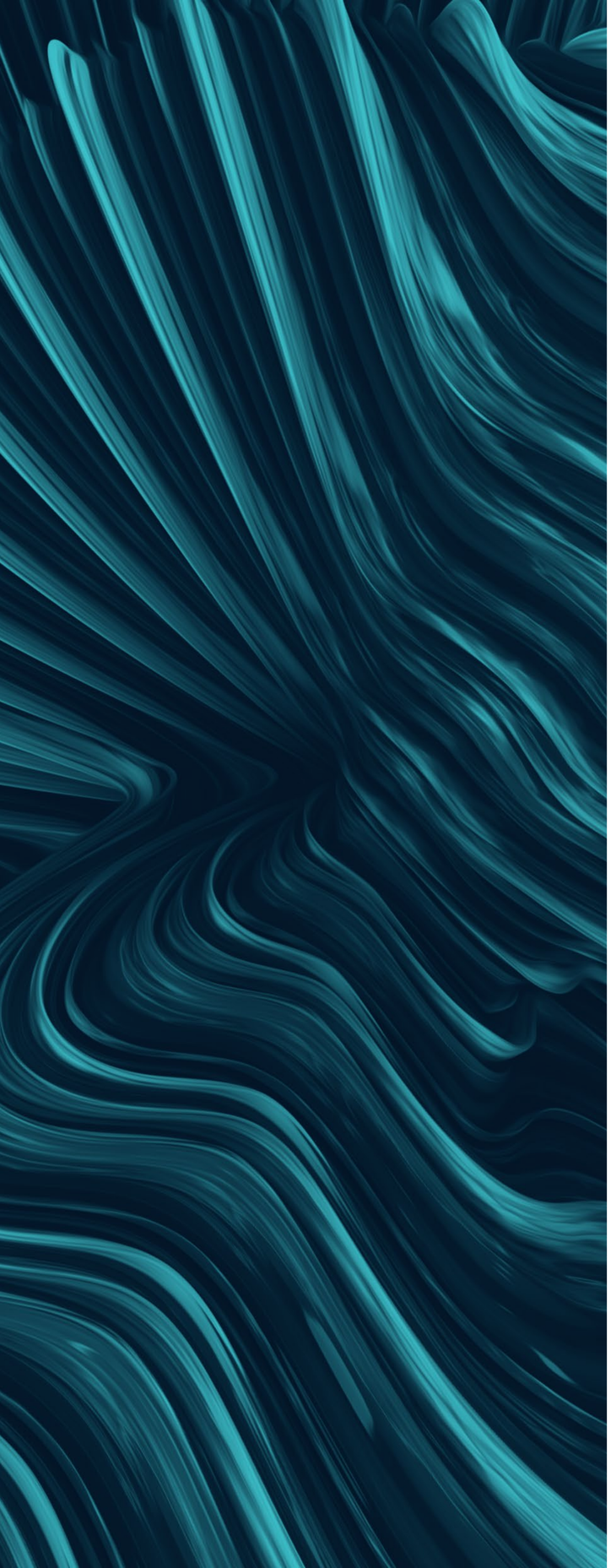
TOOL DA ADOTTARE INSIEME A KUBERNETES

Esistono strumenti che possono aiutare nell'adozione e nell'utilizzo di Kubernetes: conoscerli renderà tutto più semplice!

Il primo consiglio è di sperimentare molto con **sistemi K8s locali come Kind** o **Minikube**: "sporcarsi le mani" permette di comprenderne il funzionamento e affrontare qualche caso di troubleshooting. Fare testing dei propri software in ambiente K8s locale aiuta anche a verificarne l'effettiva portabilità e ad avere un'idea dei possibili punti critici del proprio parco software quando eseguito su Kubernetes.

Un altro strumento da conoscere sicuramente è **Helm**, **tool open source per la gestione di pacchetti**. Helm permette di semplificare e automatizzare i deployment su Kubernetes; è usato per generare i manifest, per installare l'applicazione su Kubernetes e per apportare gli aggiornamenti.





Immancabile è anche la conoscenza di **kubectl**. Si tratta della "cabina di pilotaggio" da cui **avviare qualsiasi operazione su Kubernetes e controllare i cluster**. Per tenere sotto controllo visivamente lo stato attuale del cluster e le risorse presenti, inoltre, si può utilizzare la dashboard offerta da **Lens**. Facile da consultare, ha il vantaggio di essere intuitiva e di fornire una panoramica immediata.

Ultimi, ma non per importanza, sono **i tool di monitoring e observability come Prometheus e Grafana,** o in alternativa quelli forniti dai cloud vendor. Tenere sotto controllo lo stato del sistema, il consumo di risorse ed eventuali comportamenti anomali permette di intercettare tempestivamente i problemi e rilasciare fix velocemente. Il tutto nell'ottica di prevenire possibili spese inaspettate, soprattutto nelle fasi iniziali di adozione del sistema.

CONCLUSIONE

In questa guida abbiamo mappato i principali punti di attenzione per le aziende che decidono di introdurre Kubernetes nel proprio stack IT. Una scelta da ponderare attentamente, *in primis* attraverso **un'analisi dello stato attuale** per quanto concerne tecnologia, processi e competenze interne.

Se si decide di intraprendere la strada di Kubernetes, **il processo di adozione deve seguire determinate best practice** e possibilmente procedere per step graduali. Qualunque sia la roadmap definita, è essenziale essere consapevoli della complessità dello strumento: commettere errori o leggerezze potrebbe compromettere il successo dell'operazione.

In sintesi, Kubernetes è una tecnologia dal valore inestimabile se si desideramigliorare la scalabilità, ottimizzare i costi e gestire al meglio le risorse. Rappresenta un tassello importante della digitalizzazione in ottica Cloud Native e **un vantaggio competitivo per i servizi digitali di ogni azienda**.

SPARKFABRIK