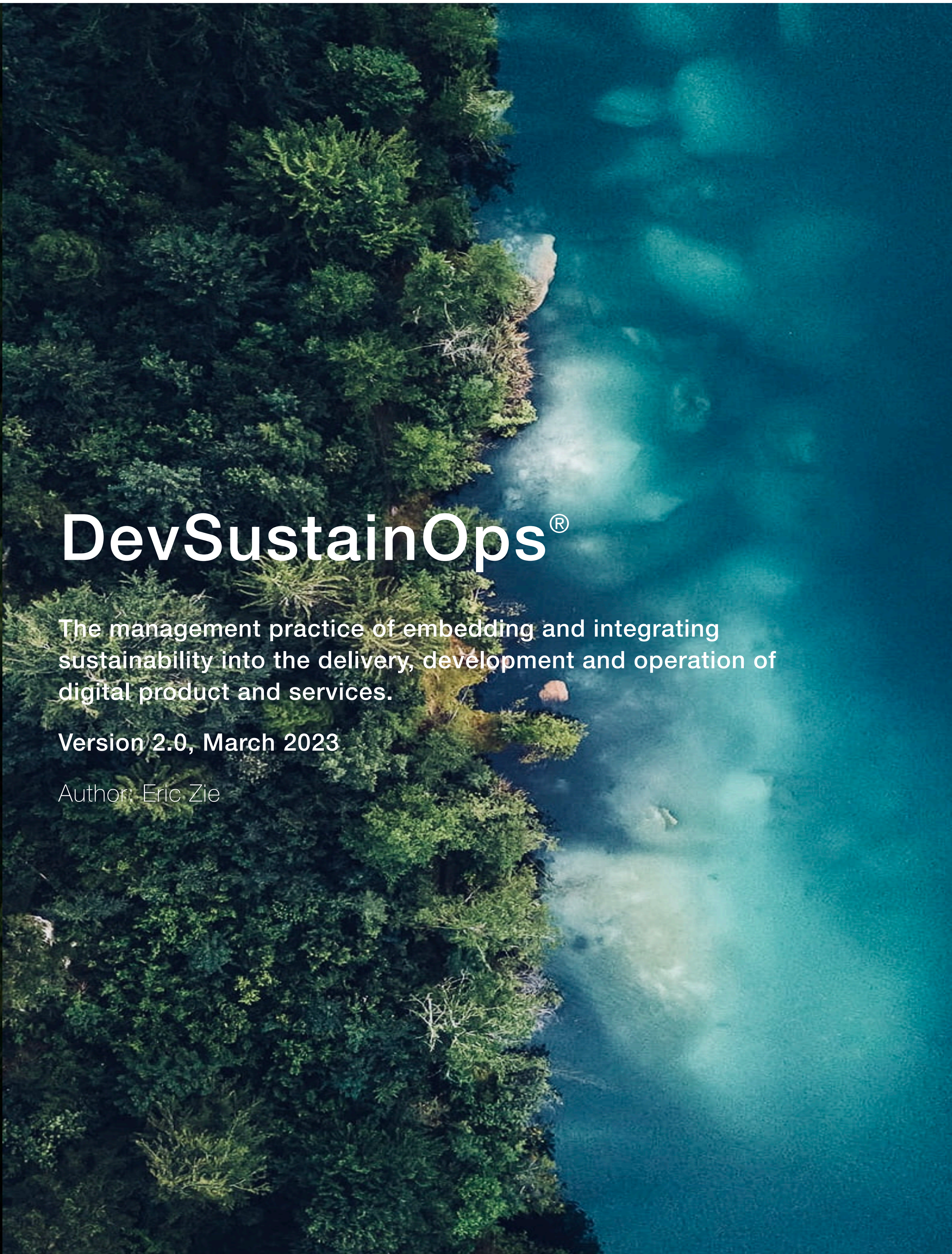
DevSustainOps



DevSustainOps®

The management practice of embedding and integrating sustainability into the delivery, development and operation of digital product and services.

Version 2.0, March 2023

Author: Eric Zie

What is DevSustainOps?

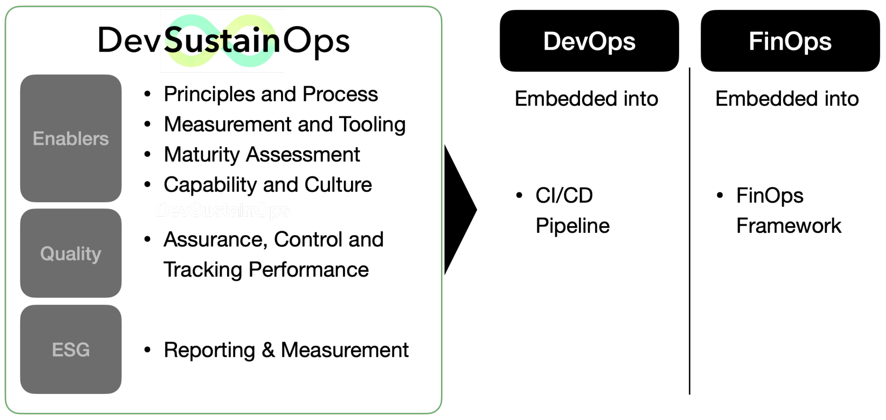
DevSustainOps is the management practice of embedding and integrating sustainability into the delivery, development and operation of software and digital product and services.

DevSustainOps is the intersection between DevOps and FinOps. The purpose is to connect and leverage all of the efficiencies, principles, best practices and automation of DevOps and FinOps to accelerate the embedding of sustainability into software development, design engineering and operation.

It is **not** intended to be a 5xOp.

It **is** intended to help practitioners rapidly enhance their DevOps and FinOps activities to help achieve a Low Carbon outcome for the digital products and services they are engineering and delivering.

DevSustainOps - Framework



The principles of DevSustainOps have been created to ensure alignment to software product life-cycle stages covering production (software, design, and engineering and quality) and use (operation and continuous improvement).

The introduction of DevSustainOps can bring Low Carbon ICT (Information and Communication Technology) and the decarbonisation of digital products and services closer to becoming a reality, and empowers those responsible for delivering new digital products to the forefront of action.

Why Does Low Carbon Software and ICT Matter?

Software runs our digital lives and businesses, but we use lots of energy to build it and need hardware, data centres and networks to keep it running. All of this consumes even more energy. The Information and Communication Technology (ICT) sector created 1.4 billion tonnes CO2e or 2.5% of global emissions in 2020 and consumed 4% of global energy – the same as all the air travel taken in the world in a year. Data centres use about 1% of global electricity – 160TWh of energy. That's the same as powering 32 million homes. And these numbers are increasing every year as our usage of digital services continues to grow.

Contributing to lowering the energy demands of software on the technology value chain and reducing the creation of millions of tonnes of CO2e now needs to be a target for all teams building software solutions. By making more efficient use of ICT it is possible to not only lower energy consumption but also extend the lifespan of the embodied carbon in all devices and infrastructure components needed to run software. By designing and building carbon efficient software it is possible to reduce total energy demands of the entire technology value chain. These actions will help us move toward a lower carbon IT future.

1. The DevSustainOps Manifesto

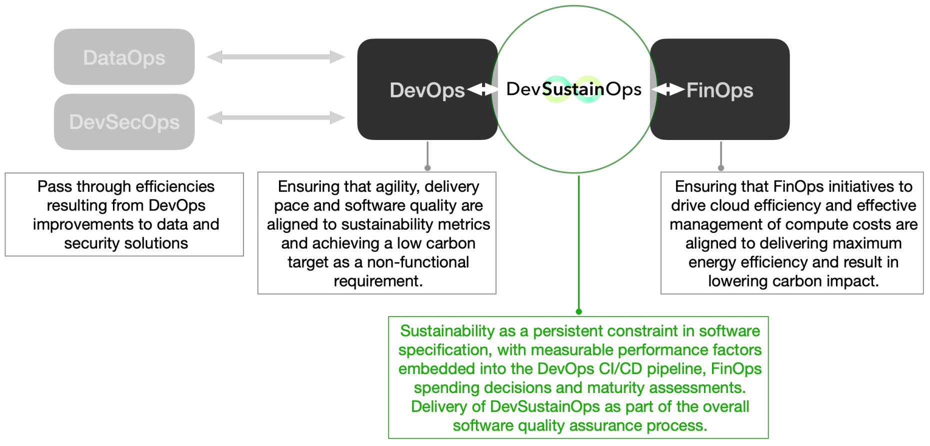
Simplicity.
Embed into DevOps and FinOps frameworks.
Immediate reduction action and impact.
Measurable.
Not a 5xOp.

These represent the core components of the DevSustainOps manifesto. The philosophy behind DevSustainOps is to reduce the energy demands of digital products and services as they are being designed, built and subsequently operated.

For ultimate success and impact, and to achieve full maturity, a persistent constraint needs to exist and to set the context for DevSustainOps to be fully integrated and successful. This context can only be achieved through the acceptance and application of a Low Carbon non-functional requirement (NFR) at the initial phase of delivery that then persists across all decisioning during delivery. Key to the manifesto is the need to measure, and this begins with an ability to establish correct tooling to allow for target setting and tracking through the delivery process.

The intention for DevSustainOps is that rather than becoming a standalone '5xOp', the principles and framework is instead integrated into the traditional 4xOps (Dev, Fin, Data and Sec) processes. The initial focus being on DevOps and FinOps but with maturity, embedding the principles into DataOps and SecOps will become a natural progression and next step.

DevSustainOps - The Intersection with DevOps and FinOps


DataOps
DevSecOps
DevOps
DevSustainOps
FinOps
Pass through efficiencies resulting from DevOps improvements to data and security solutions
Ensuring that agility, delivery pace and software quality are aligned to sustainability metrics and achieving a low carbon target as a non-functional requirement.
Ensuring that FinOps initiatives to drive cloud efficiency and effective management of compute costs are aligned to delivering maximum energy efficiency and result in lowering carbon impact.
Sustainability as a persistent constraint in software specification, with measurable performance factors embedded into the DevOps CI/CD pipeline, FinOps spending decisions and maturity assessments. Delivery of DevSustainOps as part of the overall software quality assurance process.

To summarise, DevSustainOps will seek to:

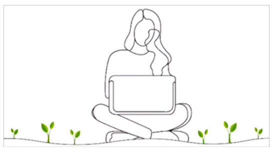
- Take the efficiencies and velocity improvements from DevOps to accelerate sustainability measures and reduce the carbon impact of digital products.
- Leverage and add to efficiencies and reduction in computing infrastructure and costs from FinOps, with resultant energy, savings and carbon reductions.
- Embed sustainability as a core quality metric in software engineering.
- Sustainability to be inserted as a non-functional requirement in agile delivery and software specification, providing a persistent constraint to effectively integrate low carbon specifications into delivery and scope.
- Pass through the resultant efficiencies from adoption and maturity to DataOps and DevSecOps practices.
- Maximise business value by helping engineering, finance, technology, ESG and business teams to collaborate on sustainability driven IT spending, prioritisation and scope decisions.
- Improve collaboration and promote continuous improvement in sustainable engineering, design and product management.
- Ensure sustainability of digital products and services can be measured in terms of maturity and performance.

2. The Benefits of DevSustainOps

By following the manifesto, the principles and objectives of DevSustainOps, it is possible to achieve the following key benefits:

The Benefits of DevSustainOps

When embedded into DevOps and FinOps processes and implemented properly with a knowledgeable cross-discipline team, DevSustainOps initiatives can improve engineering, product and infrastructure efficiencies and reduce carbon impact.

Carbon impact reduction from engineering efficiencies & shorter dev cycles	Better align software product delivery to business, ESG goals and Net-Zero commitments	Measurable carbon impact targets and KPIs
Energy aware design and build to create a sustainably minded user experience		Integrated into existing DevOps CI/CD pipeline and FinOps frameworks
Cloud resources and infrastructure utilisation optimised	Energy efficient and cost effective service improvements and operation	Quality management processes extended to track carbon performance

This can be summarised as follows:

- To begin a deeper and more complete decarbonisation of the digital products and services being designed, engineered and operated.
- By using Quality management processes the introduction of sustainability into software delivery becomes measurable and trackable, enabling appropriate prioritisation, spend decisions and performance reporting.
- Alongside decarbonisation, DevSustainOps is completely complimentary to, and will leverage and add to the operational and cost efficiencies derived from DevOps and FinOps maturity.
- And ultimately DevSustainOps also provides product management with the ability to deliver lower carbon impacting digital solutions to customers.

3. DevSustainOps Principles

To ensure that DevSustainOps activities are guided properly, the following core principles have been designed to aid practitioners. Adhering to these principles will ensure that maximum impact can be achieved when adopting and implementing DevSustainOps into your existing DevOps and FinOps frameworks.

The 6 Principles of DevSustainOps

Responsible Collaboration Engineering, operations, finance, business and ESG teams must collaborate and take responsibility for their carbon impact	**Actionable & Agile** Steps identified to reduce energy usage and lower carbon impact are actionable and achievable	**Measured** Measurement based data is the basis for informing actions and decisions to reduce carbon impact
Embedded & Planned Fully integrated into DevOps and FinOps initiatives to constantly maximise impact and value with clear execution plan	**Constant Reduction** Use CI/CD approach and efficiencies combined with FinOps cost management techniques to deliver constant carbon reduction	**Reportable** Routinely produce and share reports and performance against set carbon reduction metrics across the team

Principle One: Responsible Collaboration

A key principle in DevSustainOps is to create transparency and bring cross-functional teams together. Success cannot be the sole responsibility of the technology team, and one of the most important roles of DevSustainOps practitioners is to bring together IT professionals (including engineering, infrastructure and cloud), Business, ESG, Change and Finance, so that they can collaborate, take responsibility for, and work efficiently toward a common objective of reducing the carbon impact of the digital products and services being created.

It is also important to define a common vocabulary to develop a better understanding between the teams and to aid effective collaboration.

Adopting Low Carbon action and DevSustainOps will be is as much a cultural change as it is a technology transformation. A key aspect of DevSustainOps is to drive everyone to take accountability for the part they play to enable a reduction in carbon impact. This will help make all parties responsible for their choices and empower them to optimise their actions. It is important to strive to create a Low Carbon-conscious culture where teams are focused on innovating and boosting performance, and at the same time are aware of opportunities to optimise energy efficiencies.

Principle Two: Actionable and Agile

There are two important aspects to this principle that will make the implementation of DevSustainOps into an organisation successful:

1. Undertaking activities that are actionable and agile. The benefits of DevSustainOps are required at speed because of the nature of the challenge we are facing, i.e. we are no longer afforded the luxury of time in addressing climate change. This principle is based on the premise that insertion of DevSustainOps activities directly into existing DevOps Continuous Improvement and Delivery (CI/CD) pipelines will dramatically accelerate adoption and action. Similarly, actions aligned to the FinOps framework, will ensure that provisioning requirements and right-sizing of cloud configuration will not only result in cost efficiencies being maximised but also carbon and energy related efficiencies.

2. In addition, DevSustainOps is most successful when it is carried out in an agile manner. This means following an iterative approach with a commitment to continuous improvement and automation. It is possible to drive continuous improvement by sharing carbon-efficient, successful architectural patterns widely amongst employees. DevSustainOps recommends the application of a Low Carbon non-functional requirement into all new digital product development, ensuring that a persistent constraint is applied across all decision making, prioritisation and engineering choices.

Principle Three: Measured

Measurement is a core principle in DevSustainOps and a fundamental aspect of being able to baseline, set targets and build appropriately granular key performance indicators to track success. Similar to a successful DevOps practice requiring teams to monitor a consistent and meaningful set of DevOps KPIs to ensure that processes, pipelines, and tooling meet the intended goal of delivering better software faster. Or in the case of FinOps where practitioners undertake measuring unit costs as an objective measurement of how well an organisation is performing against not only its FinOps goals, but as a business overall.

DevSustainOps must introduce and manage low carbon metrics at the core of all activities.

In later sections the activities that require performance measurement will be explained, as will the maturity assessment grading for a team or organisation to understand their level of adoption and progress. At the core of these activities is the need to establish measurable benchmarks and metrics to monitor the carbon impact of the digital product or service being created.

DevSustainOps recommends starting by setting up standard **Low Carbon KPIs**, for example the following should be a considered a minimum standard:

- Baselining and setting targets for carbon intensity for each phase of the digital product lifecycle (at a low enough level of granularity to enable choices to affect change).
- KPIs to track variance to target through product development and use phases of the product lifecycle.
- Where multiple products form part of the end customer experience or use, the total % of product coverage being assessed to ensure completeness and impact value from a consumer perspective.
- An overall carbon impact rating for the product so that performance can be measured and understood by all parties involved.
- A DevSustainOps maturity assessment to ensure adoption progress can be measured and tracked.

Principle Four: Embedded and Planned

As previously stated, the most effective approach to DevSustainOps is to not consider it a '5xOp' and instead embed the activities directly into existing DevOps and FinOps frameworks. A prerequisite of effective DevSustainOps is therefore a well formed and reasonably mature DevOps Continuous Integration and Continuous Delivery (CI/CID) pipeline. And where cloud compute utilisation is an integral part of the product delivery, then integration of DevSustainOps activities into the FinOps framework.

Planning to embed DevSustainOps into these existing frameworks and methods facilitates adoption and accelerating the introduction of low carbon activities to promote constant carbon reduction alongside well established delivery and cloud financial management approaches. In situations where DevOps and FinOps maturity are non-existent or very low then it is essential that these are established and progressed ahead of implementing DevSustainOps.

Refer to the later section explaining how DevSustainOps activities intersect with the CI/CD pipeline and FinOps framework for more clarity.

Principle Five: Constant Carbon Reduction

The purpose of DevSustainOps is to create a culture of continuous carbon reduction alongside existing product delivery methodologies. By creating the correct culture, mindset and integrating the practice of **'Constant Carbon Reduction'** by leveraging the existing frameworks for continuous deployment in DevOps then it is possible to accelerate the delivery of sustainable digital products and services.

A target of Constant Carbon Reduction can be achieved through adopting the combined principles of DevSustainOps. The concept of 'measurement to take action' and embedding this into the approach to product delivery is key. All DevOps, FinOps and agile practitioners can, with the framework defined by DevSustainOps, take action and reduce the carbon impact of the products they design and produce, without detrimentally affecting the velocity and cost of the product being created.

Principle Six: Reportable

It is a critical principle to be able to report against the actions being taken to decarbonise a digital product or service. Having met the criteria of measuring and setting targets to achieve lower carbon impact, it is vital to be able to establish the reporting criteria and capabilities to be able to track progress.

Low carbon reporting should encompass an overview of:

- The overall status of the product delivery including the agreed low carbon measures;

- The sustainability related milestones being achieved;
- Responsibilities of each employee or team member to enable delivery against the low carbon targets set for the product;
- The issues faced by various team members to achieve the low carbon targets;
- Other important factors that affect delivering the sustainability related targets set.

Tracking and Reporting is essentially a mechanism to prevent issues before they happen, to ensure that the product will be delivered on-time, and to keep those involved informed of the product delivery progress specifically in relation to the low carbon targets set for the product.

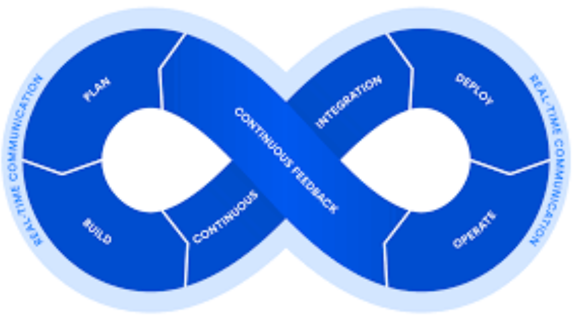
4. DevOps and FinOps Process Intersection Points

The purpose of DevSustainOps is not to create a ‘5xOp’ but instead define the principles, activities and benefits within the context of existing DevOps and FinOps frameworks. DevSustainOps needs to be measured, and quality or maturity assessed in its own right. This section aims to bring clarity to the intersection points with DevOps and FinOps and identify the activities that need to be undertaken within these frameworks to enable DevSustainOps targets to be achieved.

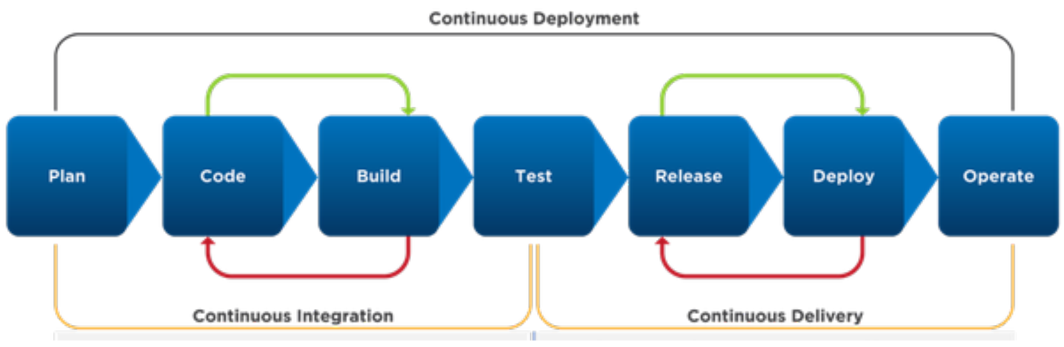
4.1 DevOps Alignment

All DevOps practitioners will be familiar with DevOps as a set of practices, tools, and a cultural philosophy that automates and integrates the processes between software development and IT teams. DevOps emphasises team empowerment, cross-team communication and collaboration, and technology automation.

Under a DevOps model, development and IT operations teams are no longer ‘siloed’. Sometimes, these two teams merge into a single team where the engineers work across the entire application lifecycle — from development and test to deployment and operations — and have a range of multidisciplinary skills. To be highly effective, DevOps teams use tools to automate and accelerate processes, which helps to increase reliability. A DevOps toolchain helps teams tackle important DevOps fundamentals including continuous integration, continuous delivery, automation, and collaboration in a continuous cycle as commonly demonstrated in the DevOps flow below:

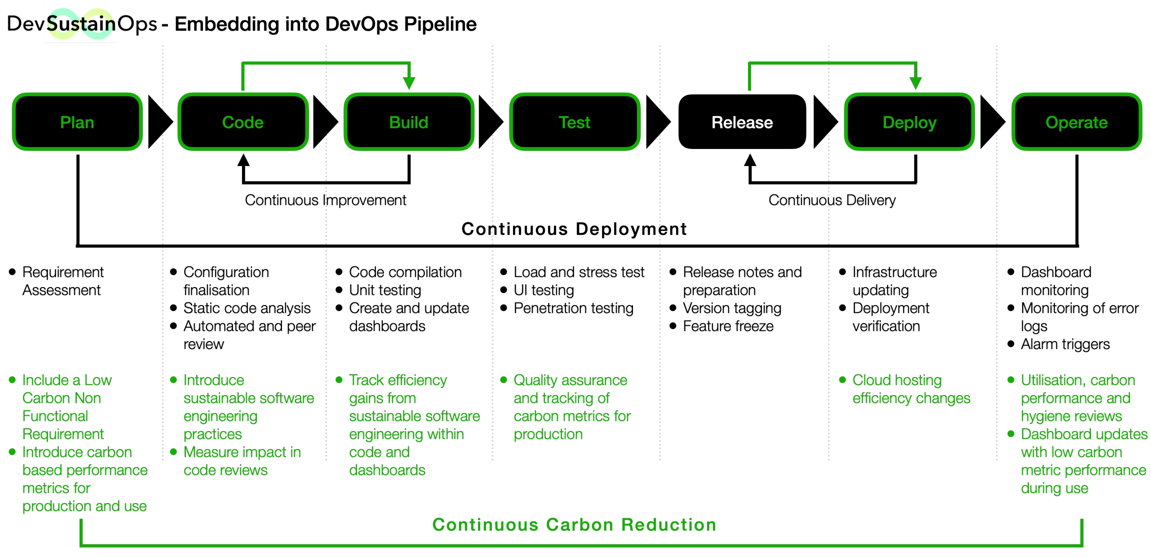


The outcome is to maximise efficiencies through automation, improve the speed and stability of software development and deployment, and help optimise the whole business. These underlying principles and benefits are closely aligned and complimentary to the objectives of DevSustainOps.



For DevSustainOps it is important to examine the DevOps continuous deployment cycle in more detail, and use the underlying DevOps pipeline framework as a basis for integrating the Low Carbon activities required to decarbonise a software product. It is by injecting the appropriate Low Carbon activities alongside existing DevOps practices without detrimentally impacting the continuous integration and delivery cycles that will make DevSustainOps straightforward to consume and action. Alignment to existing DevOps practices will also ensure that practitioners will be familiar with the structure for achieving the outcomes of DevSustainOps.

In the DevSustainOps method a set of additional activities are introduced at each stage of the continuous deployment cycle. The diagram below shows the alignment of the Low Carbon DevSustainOps activities to achieve a Continuous Carbon Reduction cycle that runs as an embedded part of the existing DevOps Continuous Deployment activities.


DevSustainOps - Embedding into DevOps Pipeline
Continuous Improvement
Continuous Delivery
Continuous Deployment
Continuous Carbon Reduction

Plan	Code	Build	Test	Release	Deploy	Operate
• Requirement Assessment	• Configuration finalisation • Static code analysis • Automated and peer review	• Code compilation • Unit testing • Create and update dashboards	• Load and stress test • UI testing • Penetration testing	• Release notes and preparation • Version tagging • Feature freeze	• Infrastructure updating • Deployment verification	• Dashboard monitoring • Monitoring of error logs • Alarm triggers
• Include a Low Carbon Non Functional Requirement • Introduce carbon based performance metrics for production and use	• Introduce sustainable software engineering practices • Measure impact in code reviews	• Track efficiency gains from sustainable software engineering within code and dashboards	• Quality assurance and tracking of carbon metrics for production		• Cloud hosting efficiency changes	• Utilisation, carbon performance and hygiene reviews • Dashboard updates with low carbon metric performance during use

The table below provides a further explanation for each DevSustainOps key activity:

DevOps Stage	DevSustainOps Activity	Description
Plan	Include a Low Carbon Non Functional Requirement (NFR)	Introduce a Low Carbon NFR alongside other non functional requirements to create a persistent constraint defined by low carbon targets through the development process.
	Introduce carbon based performance metrics for Production and Use	Specific low carbon based metrics created for the product based on a combination of customer, business and sustainability criteria and requirements. The metrics should be granular and specific, Carbon Intensity targets for a particular measure of the software product are ideal.
Code	Introduce sustainable software engineering practices	Introduce or reinforce sustainable software engineering principles and techniques into code design and development. This may require providing the engineering team with relevant training.
	Measure low carbon impact in code reviews	Begin to assess the low carbon impact of sustainable engineering techniques through measurement and tracking during code reviews. Utilise the low carbon metrics established for the product as a basis for consistent measurement. Ideally measure code performance at carbon intensity level per relevant characteristic of the product being built.
Build	Track carbon efficiency gains from sustainable software engineering within code and dashboards	Aggregate code level performance to build phase or increment of the product development. Ensure that customer and product targets are being achieved for each iteration of delivery, and provide reporting via existing dashboards. Manage all non-conformant and below target performance via existing ‘definition of done’ and acceptance criteria with reference back to the constraints placed by the Low Carbon NFR agreed at the start of the product development.

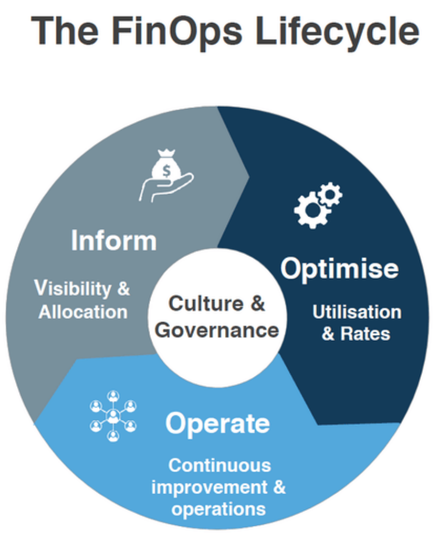
Test	Quality assurance and tracking of carbon metrics for Production (build)	Utilise the DevOps QA process to track the carbon metrics and performance against target for the production (build) stage of the product. Testing is a key part of ensuring that low carbon NFR, metrics and targets are being delivered according to agreed specification. Passing the Test phase is central to achieving the detailed sustainability targets set for the product, and a core sign off for the 'definition of done'.
Deploy	Cloud hosting efficiency changes	Implement the cloud related actions identified through measurement and action of low carbon targets. This stage will intersect with FinOps when the deployment is on Cloud. Cloud related changes may involve actions such as utilisation reviews, identification of efficiency actions, optimisation of cloud resources and continuous review and improvement.
Operate	Utilisation, carbon performance and hygiene reviews	Measure the low carbon impact during use, specifically focusing on the performance of code and the overall product during use stage. Carbon Efficiency scoring to ascertain performance against metrics and targets is essential to ensure that the sustainability of the product is being assessed accurately.
	Dashboard updates with low carbon metric performance during Use (operation)	Integrate Low Carbon metrics (carbon intensity and efficiency scores) into existing DevOps dashboards to ensure management visibility. Support corrective action where required to ensure that targets are being achieved.

4.2 FinOps Alignment

Alignment to the FinOps framework is the second intersect required to achieve DevSustainOps. FinOps (a combination of the words "Finance" and "DevOps") is a management practice that organisations use to optimise the financial performance of their cloud computing infrastructure. It promotes shared responsibility for cloud costs across information technology (IT), DevOps, and other cross-functional teams in order to improve decision making processes and drive greater business value.

The goal of FinOps is to ensure that an organisation's cloud spending aligns with its business objectives and that cross-functional teams work harmoniously to gain more financial control and predictability, reduce friction, and deliver products faster.

The three stages of FinOps are commonly identified per the diagram below and defined as:

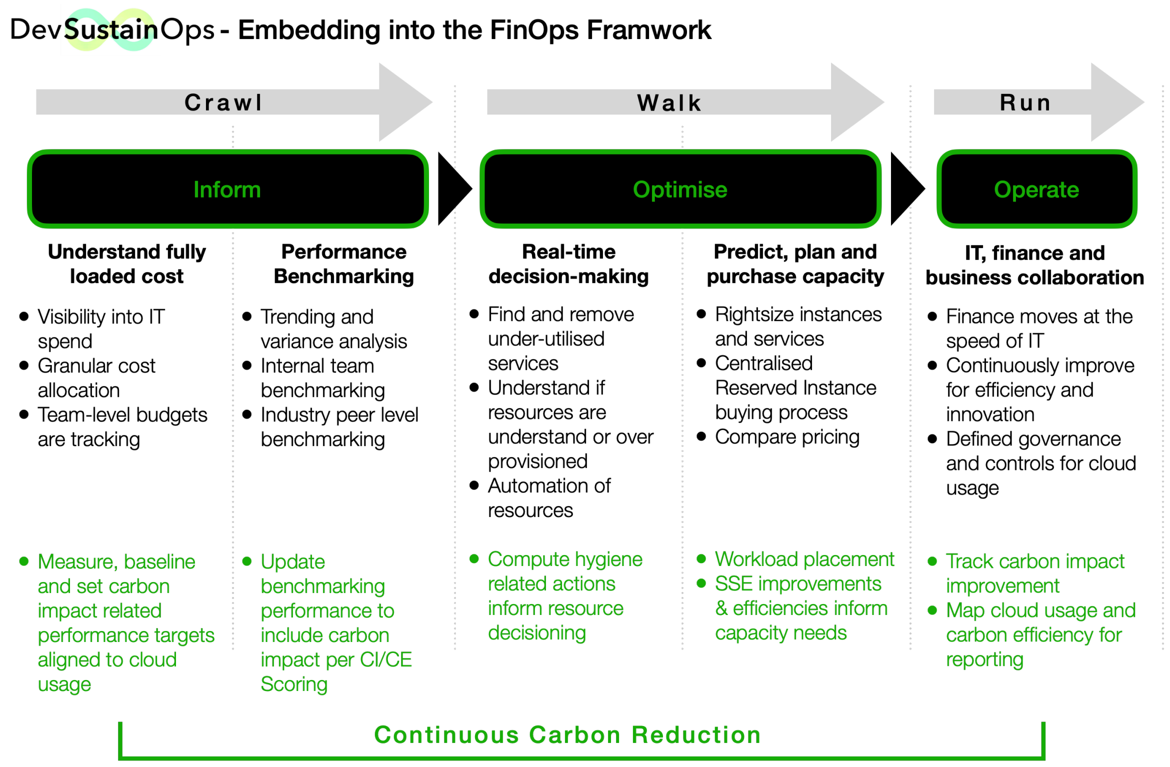


I. **Inform** - the first phase in the FinOps journey, empowering organisations and teams with visibility, allocation, benchmarking, budgeting, and forecasting. The on-demand and elastic nature of cloud, along with customised pricing and discounts, makes it necessary for accurate and timely visibility for intelligent decisions. Accurate allocation of cloud spend based on tags, accounts, or business mappings enable accurate chargeback and show back. Business and financial stakeholders also want to ensure they are driving ROI while staying within budget and accurately forecasting spend, avoiding surprises. Benchmarking as a cohort and against teams provides organisations with the necessary metrics to develop a high performing team.

II. **Optimise** - once organisations and teams are empowered, they need to optimise their cloud footprint. Cloud providers offer multiple levers to optimise. On-demand capacity is the most expensive. To encourage advanced reservation planning and increased commitment, cloud providers offer discounts for commitments which typically involves complex calculations for making reservations (Reserved Instances (RI) / Committed Use Discounts (CUD – Google Cloud). In addition, teams and organisations can optimise the environment by rightsizing and automating turning off any wasteful use of resources.

III. **Operate** - organisations start to continuously evaluate business objectives and the metrics they are tracking against those objectives, and how they are trending. They measure business alignment based on speed, quality, and cost. Any organisational success is only possible if the organisation builds a culture of FinOps which involves a Cloud Cost Center of Excellence built around business, financial, and operational stakeholders who also define the appropriate governance policies and models.

In the DevSustainOps method a set of additional activities are introduced at each stage of the FinOps lifecycle. The diagram below shows the alignment of the Low Carbon DevSustainOps activities to achieve a Continuous Carbon Reduction cycle that runs as an embedded part of the existing FinOps lifecycle stage activities:


DevSustainOps - Embedding into the FinOps Framwork
Crawl
Walk
Run
Inform
Optimise
Operate
Understand fully loaded cost
• Visibility into IT spend
• Granular cost allocation
• Team-level budgets are tracking
Performance Benchmarking
• Trending and variance analysis
• Internal team benchmarking
• Industry peer level benchmarking
Real-time decision-making
• Find and remove under-utilised services
• Understand if resources are understand or over provisioned
• Automation of resources
Predict, plan and purchase capacity
• Rightsize instances and services
• Centralised Reserved Instance buying process
• Compare pricing
IT, finance and business collaboration
• Finance moves at the speed of IT
• Continuously improve for efficiency and innovation
• Defined governance and controls for cloud usage
• Measure, baseline and set carbon impact related performance targets aligned to cloud usage
• Update benchmarking performance to include carbon impact per CI/CE Scoring
• Compute hygiene related actions inform resource decisioning
• Workload placement
• SSE improvements & efficiencies inform capacity needs
• Track carbon impact improvement
• Map cloud usage and carbon efficiency for reporting
Continuous Carbon Reduction

The table below provides a further explanation for each DevSustainOps key activity:

FinOps Stage	DevSustainOps Activity	Description
Inform	Measure, baseline and set carbon impact related performance targets aligned to cloud usage	Balance the traditional cost based approach to assessing cloud usage with the contribution of the cloud based workload and compute usage to the overall carbon cost of the product being measured. A reduction in IT spend, and visibility of the cost allocation related to the product provides a good alignment to lowering carbon impact, however it is important to identify granular carbon associated metrics for cloud usage to enable target reductions and efficiencies to be set. Work closely with the cloud provider to determine the correct level of detail and granularity and insert these metrics into the overall production and use targets for the product.
	Update benchmarking performance to include carbon impact per CI/CE Scoring	Understand the cloud carbon contribution to carbon intensity and efficiency scoring for the product. The compute related influence to a product's overall carbon impact score will always be significant. Include benchmarking comparisons for the product operating against different CSP to ensure a constant measure of carbon efficiency and performance.
Optimise	Compute hygiene related actions inform resource decisioning	Cloud based resource utilisation measurement and dynamic adjustments to maximise efficiency will also have a correlated carbon reduction impact. Ensure that cloud resource decisioning is influenced and balanced by cost as well as energy efficiency and carbon reduction.
	Workload placement	Rightsizing cloud resource provisioning to be informed by accurate placement of workloads based on energy efficiency criteria alongside other requirements. The impact of adopting SSE and resultant efficiencies should directly influence decisions to rightsize resources and achieving resultant cost efficiencies.
	Sustainable Software Engineering (SSE) improvements and efficiencies inform capacity needs	

Operate	Track carbon impact improvement	Continuous improvement driven by efficiencies and innovation should be informed by tracking the carbon reductions resulting from actions and choices made post measurement. Tracking against metrics and targets set, including those identified in the DevOps process, can result in clear reporting of improvement for cloud utilisation, cost and carbon reductions.
	Map cloud usage and carbon efficiency for reporting	Clearly map the linkage between engineering efficiency, cloud usage and carbon impact reduction in all reporting. Ensure that all stakeholders are making balanced decisions based on the reporting provided.

5. The DevSustainOps Maturity Framework

To fully embed DevSustainOps activities an understanding and close alignment to the DevOps and FinOps maturity stages is key. This will ensure all activities are interlocked and progress and mature in a synchronised manner. The ultimate aim is to reach high maturity levels across DevOps, DevSustainOps and FinOps in unison, as this will deliver maximum benefits and efficiencies. This represents one of the most significant management challenge for IT leaders - the connected and co-ordinated maturity across all three frameworks.

To begin it is important to understand the various maturity levels in both DevOps and FinOps to be able to then intersect the DevSustainOps maturity stages in the appropriate way.

5.1 DevOps Maturity Levels

DevOps maturity is based on levels that defines progress according to the following definitions:

> **Initial Level:** At the initial stages, the organisation might not be aware of DevOps or its potential benefits. Organisations at this initial or beginner level generally are characterised as following a waterfall project management approach with long approval and change processes, and teams structured around a skill for their IT projects. These teams plan and design everything up-front before the development teams start coding, and when all is done, separate teams deliver the application to production. Tests start very late in the process. Operation is a separate team that waits for developers to hand over their applications with instructions on how to deploy them.
>
> **The Repeatable Level:** At this level, organisations know the core principles of DevOps and apply them according to their daily jobs. Environments and their configurations are versioned and can be set up consistently. They are starting to collaborate effectively between development and operations. Changes are well communicated. Organisations operating at this level are not just ‘reactive’ to all that comes across their path. They are proactive and work their way towards repeatable processes for the areas they understand well. However, teams tend to ship large features

that are difficult to manage and test. Breaking bigger services into smaller micro-services remains a challenge. Operations teams frequently need to manually intervene in production to keep things running.

Defined Level: A key characteristic of this level is consistency across areas and topics. Processes are repeatable but also standardised. For example, database changes are performed automatically with every release, non-production deployments are rolled out automatically, and monitoring is integrated with every application. Integration tests are executed automatically and act as quality gates for any later stage in the delivery pipeline. Teams are organised around projects or products and not around skill-sets. Development teams work towards the execution of clear requirements that deliver clear business value. All processes are communicated clearly to all people involved, and documentation and release notes are created automatically.

Managed Level: At the managed level, all environments are managed effectively. Database changes and rollbacks are tested with every iteration of the product itself. The delivery process is predictable and runs frequently and as a result stakeholders know what and when to expect. Applications are actively monitored in production, and metrics are gathered. Teams know how to incorporate feedback for their next iteration. The organisation uses a knowledge management tool to capture existing knowledge and bring more knowledge to the teams. Mentors coach the teams to push them forward. Culture does not remain a bottleneck and welcomes change to achieve organisational goals.

Optimised Level: At the optimised level, processes are fully automated and testing is done in production. Teams know how to deal with problems like overloaded systems. The system itself will scale or adjust to peak requirements. It also adjusts to potential problems, like network interruptions or other infrastructure failures.

5.2 FinOps Maturity Levels

The practice of FinOps is iterative and maturity of any given process, functional activity, capability or domain will improve with repetition. A

"Crawl, Walk, Run" approach to performing FinOps enables organisations to start small, and grow in scale, scope and complexity as business value warrants maturing a functional activity. Taking quick action at a small scale and limited scope allows FinOps teams to assess the outcomes of their actions, and to gain insights into the value of taking further action in a larger, faster, or more granular way.

In assessing the state of an organisation's FinOps capability or domain, these maturity designations can be used to identify where a team is currently operating, and to identify areas to improve the maturity level. These terms are general guidelines, and an organisation's goal should never be simply to achieve a "Run" maturity in every capability.

As the FinOps Principles dictate, business value should drive decision making. An organisation that has established "Walk" stage anomaly detection – which has proven adequate in detecting the few cost spikes the organisation has previously experienced – should consider investing time evolving other FinOps capabilities which could provide an immediate benefit. Effort spent maturing a capability that is meeting the measurement of success could move an organisation's maturity in that individual capability from Crawl to Walk, or Walk to Run – but provide no benefit to the measurement of success.

With this in mind it is key to balance the maturity levels against the need and targets set for carbon reduction against all other requirements of the product. This balanced and pragmatic approach will lead to the best outcome, at an achievable and rapid pace.

5.3 DevSustainOps Maturity Mapping to DevOps and FinOps

Setting up the DevSustainOps maturity stages and connecting them to the DevOps and FinOps maturity models makes it possible for DevSustainOps activities to be seamlessly integrated into existing cloud management and development processes. To find out more, contact connect@gocode.green, who will be happy to discuss the framework, maturity stage descriptions, and an example of how to implement this in practice.