

Green DevOps in Practice: Using Technology for Environmental Impact

Suman Agrawal, Manoj Singh, Jyoti Mishra Parashar

Abstract

Through effective technology adoption, this article explores how DevOps methods can significantly reduce carbon emissions and environmental footprints. Teams may quantify and optimize software system energy use by including sustainability indicators into DevOps processes, from deployment and monitoring to CI/CD pipelines. Green coding concepts and techniques like cloud optimization, containerization, and automated resource monitoring reduce waste and increase energy efficiency.

Additionally, the paper presents methods for estimating the carbon footprint of software operations, offering practical advice for coordinating DevOps tactics with sustainability objectives. Through the integration of environmental responsibility into DevOps, this study demonstrates how technology can spur innovation and promote environmental sustainability.

1. Introduction

DevOps, the synergy of development and operations, revolutionizes software delivery with speed, reliability, and quality at its core. In an era where sustainability is a major demand, DevOps has the potential to drive environmental responsibility alongside innovation. By adopting energy-efficient tools, optimizing the technology workflow, and embedding sustainable practices into the DevOps lifecycle, the organizations can reduce the carbon emissions and resource consumption. This paper explores practical strategies to align DevOps with the goals of sustainability, demonstrating how technology can contribute to a greener planet.

2. The Role of DevOps in Sustainability

2.1 Alignment with Green Infrastructure and Software Engineering Practices.

Green DevOps focuses on designing and operating the software development lifecycle with minimal environmental impact while maintaining optimal application performance. DevOps teams can embrace these principles by:

- Provisioning energy-efficient infrastructure and automating its management.
- Embedding sustainability metrics into CI/CD pipelines to track and reduce resource usage.
- Educating teams on green coding practices to create energy-efficient applications.

2.2 Carbon Awareness in DevOps Practices

Adopting carbon-aware practices enables DevOps teams to integrate sustainability into their operations effectively. This includes:

- Monitoring carbon footprints and automating resource optimization based on real-time carbon intensity data.
- Automating scaling decisions to reduce carbon emissions during non-peak periods.
- Incorporating maintenance routines that ensure software remains energy-efficient and sustainable over time.

3. Infrastructure Procurement & Setup

Migrating on-premises systems to the cloud can significantly reduce energy consumption, with potential savings of 50% to 60%.

3.1 Choosing Green Cloud Regions

Green cloud regions refer to data center locations operated by cloud service providers (e.g., AWS, Azure, Google Cloud) that prioritize sustainability through the use of renewable energy sources, energy-efficient infrastructure, and environmentally friendly practices. These regions are designed to minimize carbon emissions and environmental impact while maintaining high performance and reliability for cloud services.

By provisioning our servers in the green regions we can reduce Carbon Footprint, Comply with Sustainability Goals and have lower operational costs.

Some examples of the green regions:

Cloud Provider	Green Region	Location	Sustainability Initiatives
Amazon Web Services (AWS)	US West (Oregon)	Oregon, USA	AWS has invested in wind and solar farms across the U.S., including in Oregon, to power its data centers with renewable energy.
Microsoft Azure	West Europe	Netherlands	Azure's West Europe region utilizes renewable energy sources and has implemented energy-efficient infrastructure to reduce carbon emissions.
Google Cloud Platform (GCP)	europe-west6	Zurich, Switzerland	Google has partnered with Green.ch to utilize renewable energy for its Zurich data center, aiming for carbon-free energy operations.

3.2 Instance and Service Optimization

To enhance energy efficiency in cloud environments, prioritize the use of virtual machine (VM) families with energy-efficient processors, such as Graviton processors. For example, M7g, C7g, and R7g server families reduce energy consumption by 60% while offering 40% better price performance. Additionally, leveraging managed cloud services like AWS EMR and RDS, instead of manual installations, optimizes energy usage and operational efficiency.

3.3 Leveraging Autoscaling and Lifecycle Management

Implement smaller instances for running applications and utilize autoscaling to dynamically adjust resources based on demand, ensuring high availability while reducing unnecessary compute resources. Automate storage lifecycle management to remove stale or unused data, minimizing overhead and resource consumption, thereby lowering the overall environmental impact.

4. Resource Optimization Strategies

4.1 Automating Idle Resource Shutdown

Automate the shutdown of staging servers during non-business hours, weekends, and holidays through scripts or scheduled tasks. This helps reduce unnecessary energy consumption and operational costs while ensuring that resources are only used when needed.

4.2 Right-Sizing and Resource Consolidation

Choose instance sizes that are closely aligned with application requirements to avoid overprovisioning. Continuously monitor resource usage patterns and adjust by downsizing underutilized volumes, ensuring more efficient use of resources and reducing waste.

4.3 Efficient Data Management

Optimize storage by compressing stored data and eliminating redundant datasets. For applications requiring shared VM storage, use solutions like EFS rather than EBS. Additionally, automate the cleanup of unattached EBS volumes and obsolete builds or artifacts to minimize unnecessary resource consumption and waste.

5. Network Optimization

5.1 Regional Considerations

Provisioning servers in regions closer to end-users reduces data travel distances, lowering latency and energy consumption. This strategy improves performance and enhances resource efficiency by minimizing the energy required for data transmission.

5.2 Caching and Replication

Using edge locations to cache static content speeds up response times and reduces load on central servers. Implementing read replicas for read-heavy operations helps minimize latency and network congestion, improving overall system performance.

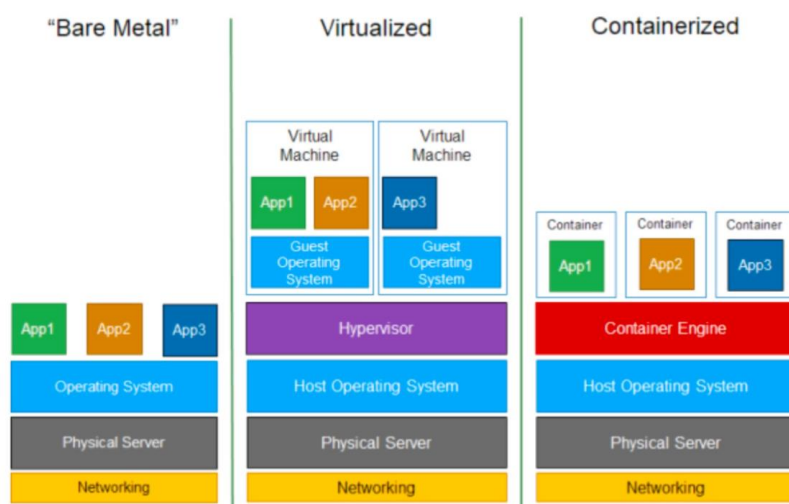
5.3 Trusted Advisor and Monitoring Tools

AWS Trusted Advisor helps identify unused resources and suggests optimizations to improve efficiency and reduce costs. Regular system monitoring ensures that performance is continually assessed, allowing for ongoing improvements in resource utilization and energy efficiency.

6. Advanced Strategies for Sustainable DevOps

6.1 Virtualization and Containerization

Virtualization helps achieve sustainability by consolidating workloads onto fewer physical servers, reducing hardware requirements and energy consumption. It enables efficient resource utilization, minimizing waste and carbon footprints. Fewer physical servers lead to lower cooling and power needs, making data centers more energy-efficient. Additionally, virtualization supports cloud solutions and remote work, further reducing commuting-related emissions. Overall, it optimizes infrastructure and contributes to a more sustainable IT ecosystem.

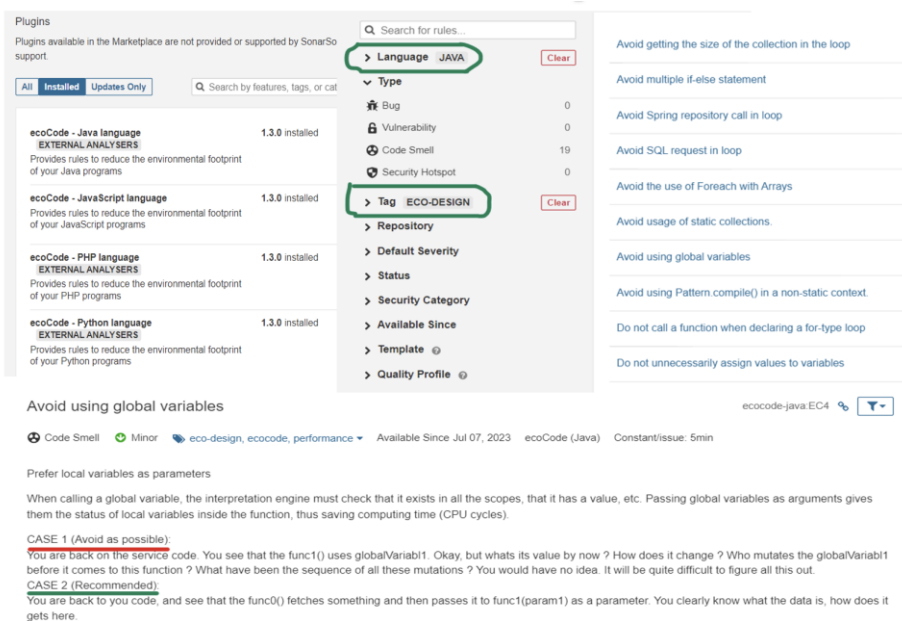


Containerization through technologies like Docker and Kubernetes significantly contributes to sustainability by optimizing resource utilization and reducing environmental impact. By allowing multiple applications to run on the same server, containerization maximizes hardware efficiency, which reduces the need for additional physical resources. Containers are lightweight and efficient to deploy, minimizing overhead and energy consumption. Their portability across different environments also helps reduce infrastructure requirements. Additionally, containerization enables the reuse of older hardware, further decreasing e-waste. The scalability of containers allows resources to be dynamically allocated based on demand, ensuring minimal waste and optimizing overall system performance.

6.2 Green Ci/CD Pipelines:

Adopting green DevOps practices throughout the software development lifecycle (SDLC) and DevOps pipeline helps optimize resource and server usage, contributing to the sustainability of the final product. Key strategies include:

Integrating the **SonarQube EcoCode plugin** in our **CI/CD** pipeline to check code quality and give eco recommendations to write sustainable code.



Instead of triggering the pipeline on every small commit made by developer in their feature branch, the process to be streamlined to trigger the pipeline only when code has been reviewed and merged to a specific branch or prefer manual trigger. Thus, by streamlining CI/CD pipelines by minimizing the frequency of pipeline runs per day to reduce resource consumption.

Similarly implementing conditional execution for **regression test cases** and Sonar scans, triggered only, when necessary, based on application requirements.

When it comes to Source code repositories like **git** using shallow clones instead of full clones during version control operations to reduce resource usage, faster clone time, improved performance.

To clone only the latest commit of a repository, you can use the following command:

```
git clone --depth 1 <repository_url>.
```

If you need more than the latest commit, you can specify a different depth:

```
git clone --depth 10 <repository_url>
```

Transitioning to cloud-based CI/CD tools such as AWS Code Pipeline, GCP Cloud Build, or Microsoft Azure Pipelines, which works on **serverless architecture**, is a more sustainable way to run the DevOps pipelines.

6.3 Go Serverless:

Adopting serverless computing significantly advances sustainability by optimizing resource utilization and reducing environmental impact. With serverless, resources are allocated only when needed, ensuring infrastructure isn't left running idle and minimizing energy consumption. Automatic scaling based on demand ensures efficient use of compute resources, preventing over-provisioning and reducing unnecessary waste. By eliminating the need for physical servers and virtual machines, serverless computing cuts down on hardware requirements and e-waste. Leading cloud providers like AWS Lambda and Google Cloud Functions enable serverless architectures by offering event-driven compute services that automatically scale, further enhancing efficiency and sustainability. These platforms leverage advanced, energy-efficient data centers, often powered by renewable energy, reducing the carbon footprint. The pay-per-use model ensures that businesses only consume energy for active workloads, promoting cost efficiency and contributing to a greener, more sustainable IT ecosystem. Overall, serverless computing drives smarter, more efficient energy use, empowering organizations to scale applications while minimizing their environmental impact.

6.4 Backup and Retention Policies

Efficient data management is a critical aspect of sustainable DevOps practices, as it helps minimize storage resource consumption and reduces the environmental impact of data operations. Key strategies include:

- **Define Retention Policies and Set Clear Objectives:** Establishing clear retention policies is essential for optimizing storage and minimizing unnecessary resource usage. Define specific objectives for Recovery Time Objectives (RTO), Recovery Point Objectives (RPO), and Mean Time to Recovery (MTTR) to ensure that backups are managed effectively while meeting business continuity requirements.
- **Backup Cleanup and Compression:** Implement automated processes to clean up and compress backups based on retention policies. This includes managing snapshots, Amazon Machine Images (AMIs), and logs, ensuring that only essential data is retained. Regular cleanup helps reduce the need for excessive storage resources, cutting down on energy consumption and optimizing system performance.
- **Versioning Policies for Data Management:** Establishing clear versioning policies helps manage data efficiently and avoids unnecessary duplication. Proper version control ensures that only the latest relevant data is maintained, reducing storage overhead and improving resource allocation across the DevOps pipeline.

By integrating these practices, organizations can achieve a more sustainable approach to data management, reduce storage-related resource waste, and enhance overall operational efficiency.

7. Sustainability Monitoring Tools.

Monitoring existing resource utilization and optimizing based on the metrics provided by various monitoring tools is crucial in ensuring progress toward sustainability goals. Several tools are available to help track carbon footprints, enabling more informed decisions on resource management. We have also leveraged open-source tools such as Scaphandre and Powertop to monitor resource usage effectively. These tools provide valuable insights that guide

us in optimizing our code and utilizing resources more efficiently, ultimately supporting our sustainability initiatives.

Infrastructure and Servers Monitoring tools:

Infrastructure monitoring tools track resource utilization of applications, providing insights into how resources are consumed across systems. Here are a few examples:

Host and Network Monitoring via AWS Cloudwatch



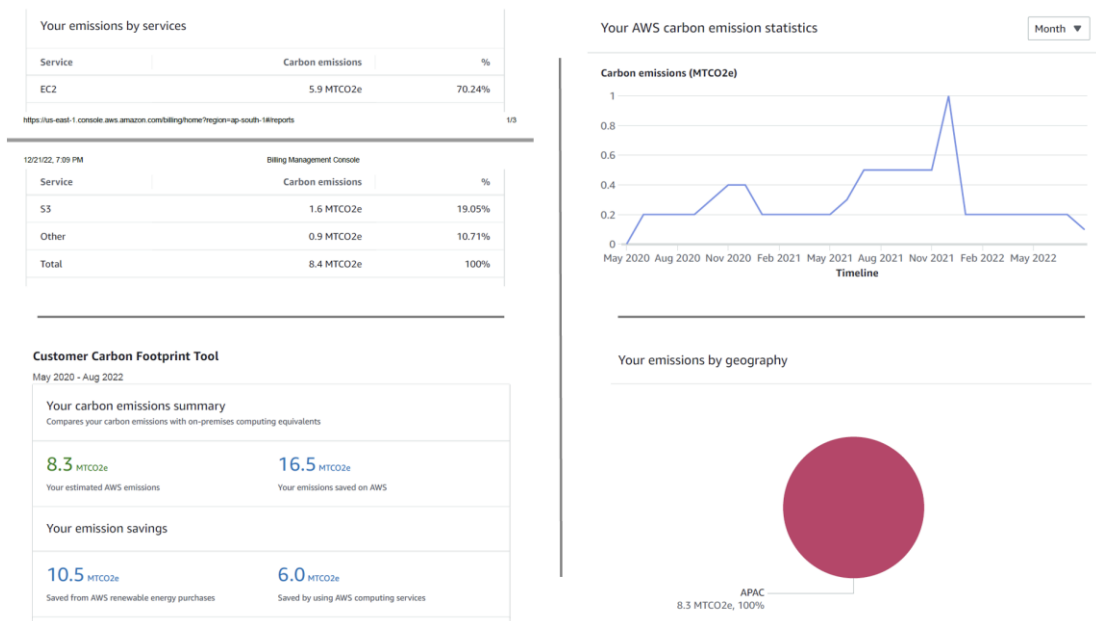
Containers Monitoring via Grafana



Carbon Footprint Tools:

Tools like AWS Carbon Footprint Tool, Azure Emissions Dashboard, and Google Cloud Carbon Footprint to measure and optimize environmental impact.

Carbon Footprint measurement via AWS Carbon FootPrint Tool



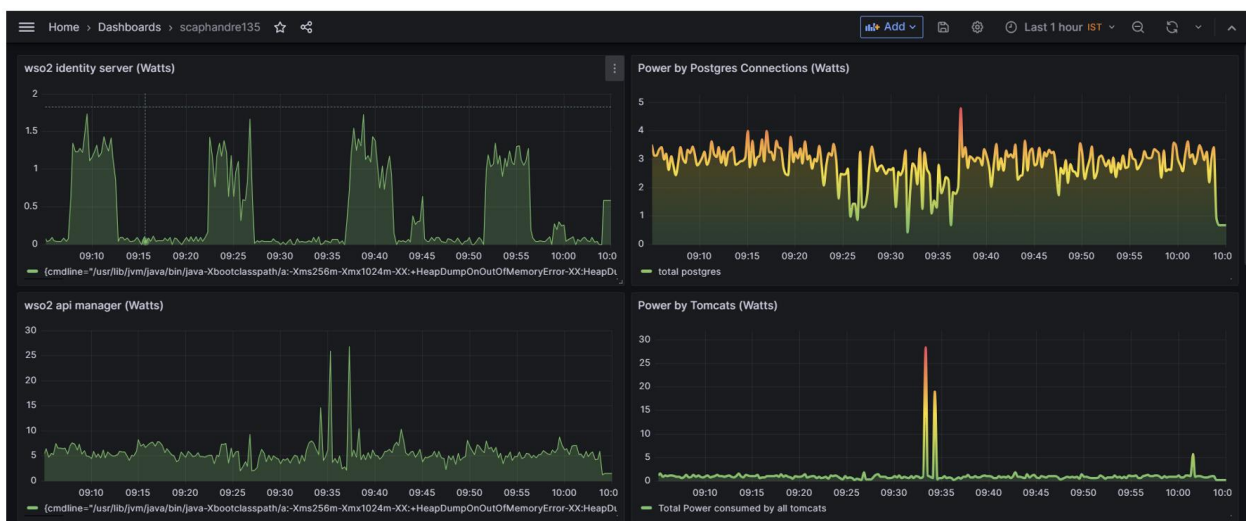
Open-source tools such as Carbon Calculator, CodeCarbon, and Kepler provide actionable insights for sustainability goals.

Power usage Monitoring via POWERTOP

PowerTOP 2.14			
Overview	Idle stats	Frequency stats	Device stats
Summary: 2490.5 wakeups/second, 0.0 GPU ops/seconds, 0.0 VFS ops/sec and 11.4% CPU use			
Usage	Events/s	Category	Description
0.9 ms/s	473.2	Timer	tick_sched timer
22.2 ms/s	158.4	Process	[PID 1147536] powertop
1.8 ms/s	103.7	Timer	hrtimer wakeup
278.0 μs/s	92.1	Process	[PID 13] [rcu_sched]
362.7 μs/s	72.9	Process	[PID 6607] /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
0.8 ms/s	48.0	Process	[PID 40809] /usr/bin/vault server -config=/etc/vault.d/vault.hcl
0.9 ms/s	47.0	Process	[PID 33138] /usr/bin/vault server -config=/etc/vault.d/vault.hcl
1.0 ms/s	43.2	Process	[PID 1541] /usr/libexec/platform-python -Es /usr/sbin/tuned -l -P
279.8 μs/s	43.2	Process	[PID 33128] /usr/bin/vault server -config=/etc/vault.d/vault.hcl
1.0 ms/s	37.4	Interrupt	[3] net_rx (softirq)
2.5 ms/s	33.6	Process	[PID 1079] /opt/microsoft/mdatp/sbin/mdatp_audisp_plugin
5.0 ms/s	29.8	Process	[PID 1988] /opt/microsoft/mdatp/sbin/mdatp_audisp_plugin

Scaphandre: Energy Consumption Metrology Agent

Scaphandre is a powerful tool designed to dive deep into your operating system, collecting detailed power consumption metrics. It utilizes the RAPL (Running Average Power Limit) sensor to gather accurate energy usage data, which is then exported through an exporter. The exported data can be integrated into Grafana dashboards, providing clear visualizations for monitoring and optimizing energy consumption across your system.



8. Conclusion

Green DevOps provides a framework for embedding sustainability into software development and operations. By adopting green cloud strategies, optimizing resources, and embracing carbon-aware standards, organizations can reduce their environmental impact while maintaining efficiency and scalability. This approach aligns technical goals with broader corporate sustainability objectives, ensuring a more sustainable future for software development.

9. Future Work

Further research is needed to:

- Develop standard metrics for measuring carbon footprints in DevOps pipelines.
 - Explore AI-driven optimizations for sustainability in cloud operations.
 - Integrate renewable energy forecasts into infrastructure scaling decisions.
-

References

- <https://www.thegreenwebfoundation.org/support/i-use-amazon-web-services-aws-am-i-a-green-site-service/>
- <https://cloud.google.com/sustainability/region-carbon>
- <https://learn.microsoft.com/en-us/industry/sustainability/availability>
- <https://towardsdatascience.com/how-to-measure-the-carbon-footprint-using-vertex-ai-pipelines-3d6bc9695e7b>
- <https://pypi.org/project/carbon-calculator/>
- <https://github.com/mlco2/codecarbon>