Microsoft

Sustainable Software Engineering Patterns

Torsten Stiller
Developer Lead at Microsoft,
Azure BG - Germany

“What pleases me most is that sustainable development is on almost everybody’s agenda now.”

Maurice Strong

(*1929 †2015) - former Director United Nations Environment Programme

Agenda

Green Software Definition

Explanation of terms

Design Areas

Patterns

Resources

Green Software Definition

- **Green Software** is software that is responsible for emitting fewer greenhouse gases.[1]

  → Use less physical resources (hardware)

  → Use less energy

  → Use low carbon energy (carbon awareness)

1 https://greensoftware.foundation/articles/what-is-green-software


CO_2
Green Software Foundation
greensoftware.org
creative commons

Green Software Foundation

Principles Alignment

Carbon Efficiency: Emit the least amount of carbon possible.

Energy Efficiency: Use the least amount of energy possible.

Carbon Awareness: Do more when the electricity is cleaner + do less when the electricity is dirtier.

Hardware Efficiency: Use the least amount of embodied carbon possible.

Measuring: What you can't measure, you can't improve.

Climate Commitments: Understand the exact mechanism of reduction.

Explanation of terms

- **Carbon footprint** is the total greenhouse gas (GHG) emissions caused directly and indirectly by an individual, organization, event or product[1]
- **Carbon dioxide equivalent** (CO^2e) is a unit used to express the carbon footprint of all greenhouse gases together as if they were all emitted as carbon dioxide
- **Carbon intensity** of electricity is a measure of how much CO^2 emissions are produced per kilowatt hour of electricity consumed.

1 The Carbon Trust (2018) Carbon Footprinting

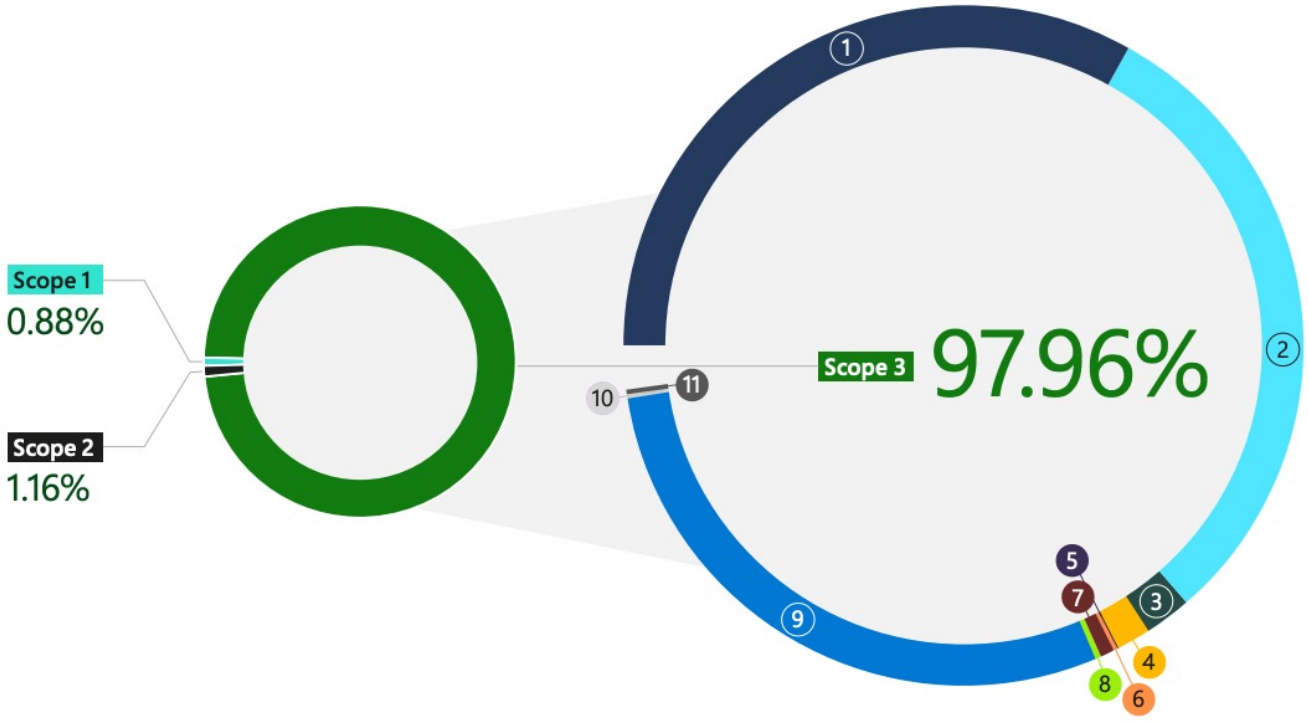
Explanation of terms

- **PUE (Power Usage Effectiveness)** is an indicator for measuring the energy efficiency of a datacenter. PUE is determined by dividing the total amount of power entering a data center by the power used to run the IT equipment within it
- **WUE (Water Usage Effectiveness)** is a metric to measure datacenter sustainability in terms of water usage and its relation to energy consumption.
- **Scope 1 emissions** are direct emissions from company-owned and controlled resources
- **Scope 2 emissions** are indirect emissions from the generation of purchased energy
- **Scope 3 emissions** are all indirect emissions - not included in scope 2 - that occur in the value chain of the reporting company, including both upstream and downstream emissions.

Explanation of terms

Breaking down of our FY21 Scope 3 emissions by source

Scope 3 represents the majority of Microsoft's emissions, and we are committed to reducing these emissions by more than 50 percent by 2030. Tracking and reporting against this category of emissions is critical for net zero progress.



Scope 3 Categories

1	Purchased Goods & Services	35.03%
2	Capital Goods	29.70%
3	Fuel- and Energy-Related Activities (Market-Based)	2.20%
4	Upstream Transportation	1.60%
5	Waste	0.04%
6	Business Travel	0.16%
7	Employee Commuting	0.57%
8	Downstream Transportation	0.32%
9	Use of Sold Products	28.07%
10	End of Life of Sold Products	0.14%
11	Downstream Leased Assets	0.13%

See appendix for more data on our carbon emissions

Explanation of terms

- **Software Carbon Intensity (SCI) score** can be used to measure and reduce carbon footprint of applications

SCI = (E*I)+M

Where: E = Energy, I = Carbon Emissions, M = Embodied Carbon

"If you can't measure it, you can't improve it." - Peter Drucker

Check out the Software Carbon Intensity (SCI) Specification here

Measurement Tools
When developing locally the most accurate is a *Watt Meter*
Other Software Utilities: *Windows E3 Tool, Intel's VTune Profiler*

Facts

Data centers are currently responsible for **1% of global energy usage**

Global trends in digital and energy indicators, 2015-2021

	2015	2021	Change
Internet users	3 billion	4.9 billion	+60%
Internet traffic	0.6 ZB	3.4 ZB	+440%
Data centre workloads	180 million	650 million	+260%
Data centre energy use (excluding crypto)	200 TWh	220-320 TWh	+10-60%
Crypto mining energy use	4 TWh	100-140 TWh	+2 300-3 300%
Data transmission network energy use	220 TWh	260-340 TWh	+20-60%

Sources: Internet users [ITU (2022)]; internet traffic [IEA analysis based on Cisco (2015); TeleGeography (2022); Cisco (2019), Cisco Visual Networking Index]; data centre workloads [Cisco (2018), Cisco Global Cloud Index]; data centre energy use [IEA analysis based on Malmodin & Lundén (2018); ITU (2020); Masanet et al. (2020); Malmodin (2020); Hintemann & Hinterholzer (2022)]; cryptocurrency mining energy use [IEA analysis based on Cambridge Centre for Alternative Finance (2022); Gallersdörfer, Klaaßen and Stoll (2020); McDonald (2022)]; data transmission network energy use [Malmodin & Lundén (2018); Malmodin (2020); ITU (2020); Coroama (2021); GSMA (2022)].

What is Microsoft doing as platform provider?

- Liquid immersion cooling, grid-interactive UPS batteries, underwater datacenters
- Two phase immersion cooling reduced power consumption for any given server by 5% to 15%.
- Unlike legacy lead acid batteries, grid-interactive UPS batteries store energy at up to 90% efficiency

Our Project Natick team is **testing an underwater datacenter in Scotland**

The project uses cold seawater to cool servers **without tapping freshwater resources, and with greater cooling efficiency than air**

Underwater servers failed at one-eighth the rate of a land-based control group, **reducing server waste**

Microsoft Azure

Netherlands

Sustainability fact sheet

As more people and businesses rely upon technology to stay connected, informed, and productive, digital needs in the Netherlands and around the globe are growing and that means the need for datacenters is growing too.

The Microsoft Cloud offers customers an energy efficient and carbon neutral alternative to running their own private datacenters. Research shows that Microsoft Cloud services can be up to 93 percent more energy efficient than traditional enterprise datacenters.

We're committed to providing a sustainable Microsoft Cloud, so we wanted to share information about how we take responsibility for our datacenter operations.

For Microsoft's datacenters in the West Europe region, located in the Netherlands, we have included local sustainability investments and datapoints in support of meeting and exceeding our commitments around carbon, water, waste, and ecosystems.

Published September 2022. This document shares information we have as of the publication date, and it includes estimated information and projections. The information is provided as-is and may change without notice.

Carbon

Power usage effectiveness (PUE)

1.16

12-month weighted average through February 2022

Design average PUE **1.12**

Microsoft's commitment

Microsoft pledged to become carbon negative by

2030

And by 2050, removing historical carbon since its founding in 1975

91% renewable energy coverage from wind power

approximate energy procured through June 2021

Microsoft will reduce Scope 1 and 2 emissions to near zero through energy efficiency work and by reaching

100% renewable energy supply by 2025.

In the Netherlands, we plan to use a **renewable biofuel blend to power our backup generators that reduces net carbon emissions.**

New Microsoft datacenters are designed to earn **LEED Gold certification.**

Microsoft has also committed by **2030** to be:

Diesel free

100% of electricity consumption, 100% of the time, matched by zero carbon energy purchases

Reducing our Scope 3 emissions by more than half

Learn about PUE and WUE

Water

Water usage effectiveness (WUE)

0.08 $\frac{L}{kWh}$

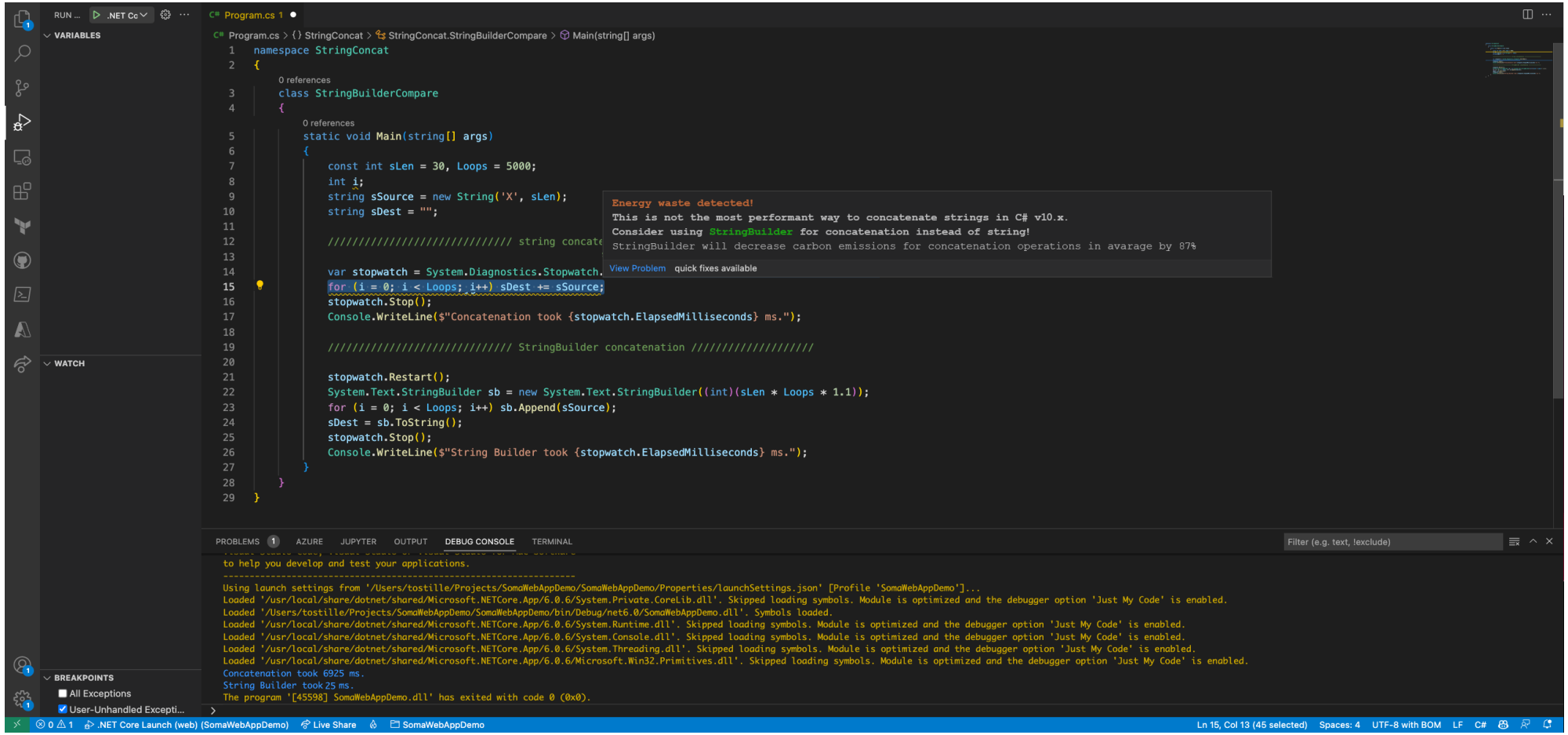
January 2021 – December 2021

In 2020, Microsoft pledged to be water positive for our direct operations by 2030.

Through this commitment, we will replenish the water consumed by datacenter operations in water-stressed regions. We have also committed to reduce water waste in our datacenter operations by 95% by 2024.

Check out more Datacenter fact sheets here

What is Microsoft doing as developer enabler?



Design Areas

UI/UX Design

Application Design

Coding

Platform

Culture

UI/UX Design
Patterns

- Appropriate device specific image sizing & number of images
- Sensible image & video resolution and codec usage
- Prevent screen flooding with preload/autoplay videos
- Intelligent user-centric visual design - avoid browsing
- Use vector graphics whenever possible

LEARN TV

Watch live and recorded events

View streaming technical content about Microsoft products from the experts that build and use it every day.

Start watching now

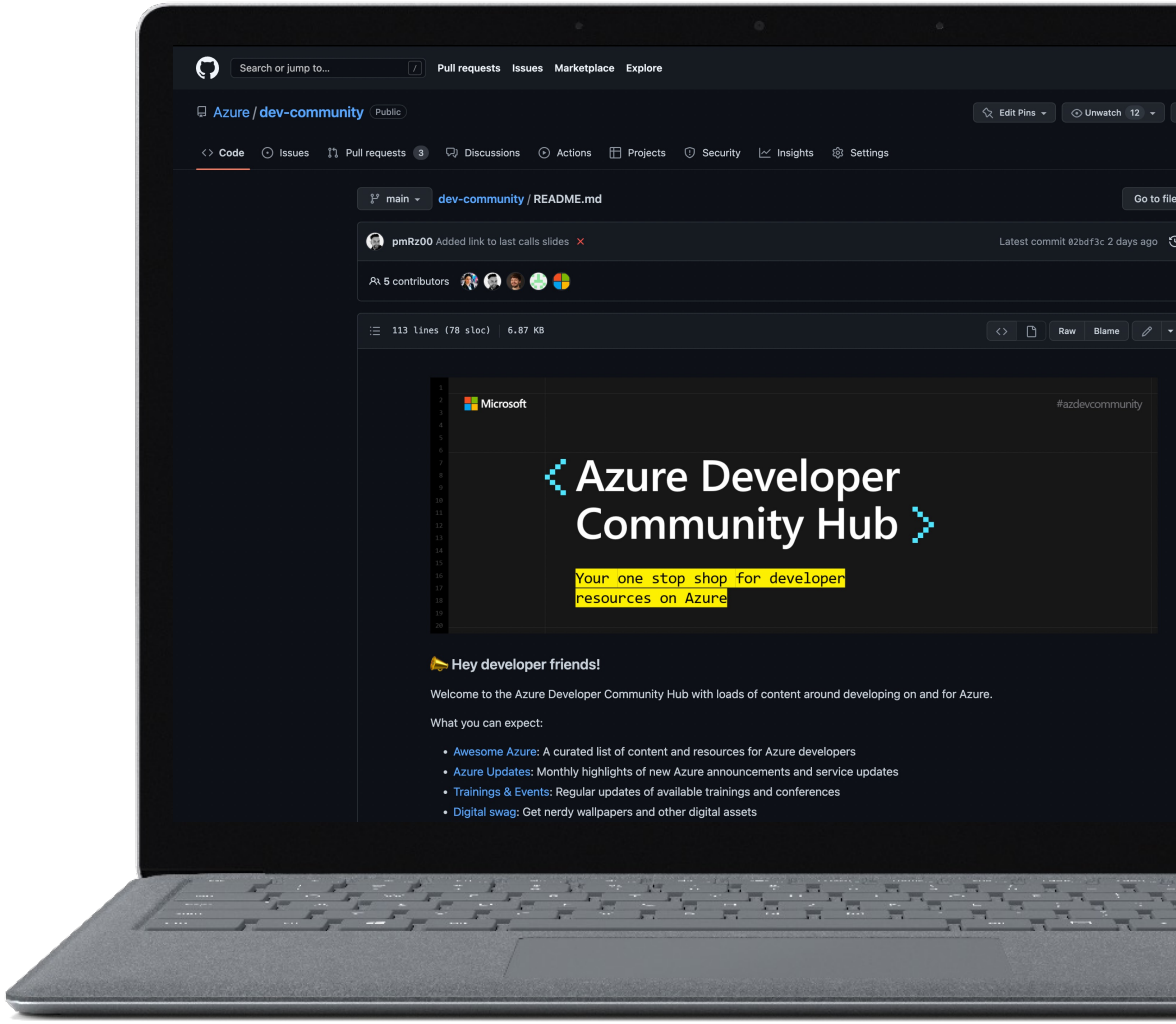
Learn TV

Video content by developers and technical enthusiasts devoted to including you in the conversation.

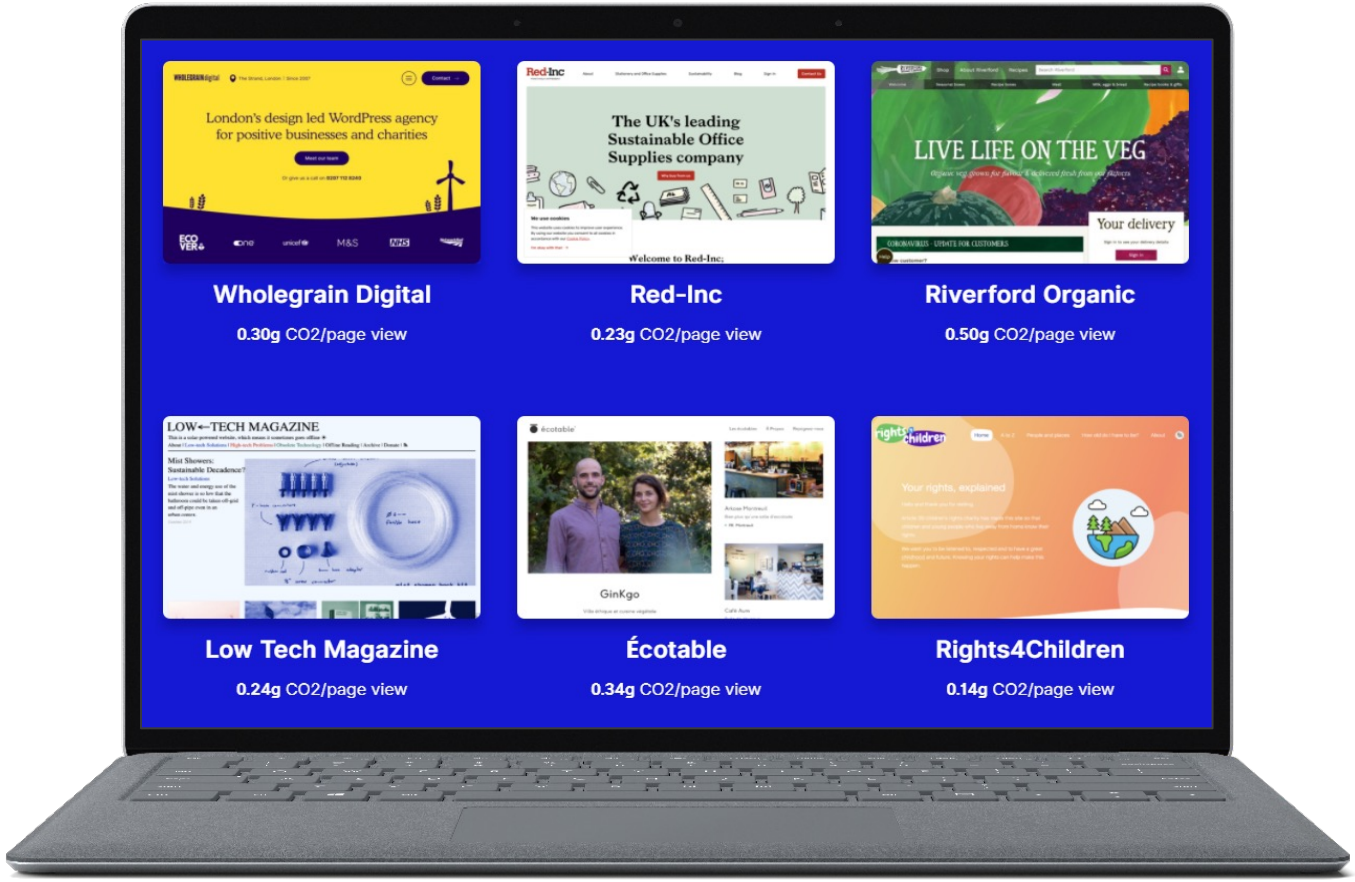
UI/UX Design

Patterns

- Consider offering dark mode
- Consider using system fonts
- Use responsive web design concept
- Sensible server update response time
- Evaluate server-side vs. client-side rendering

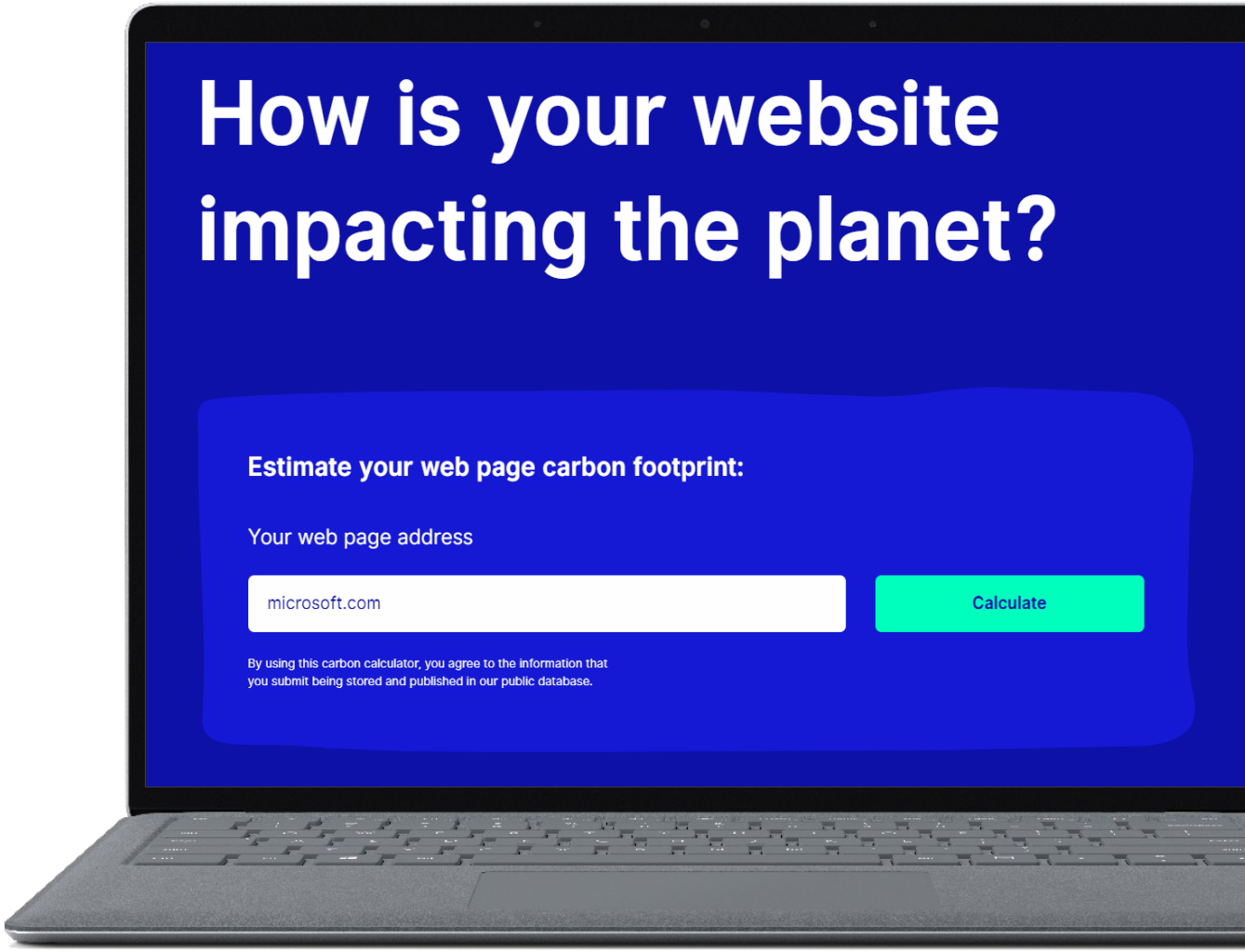


UI/UX Design
Patterns

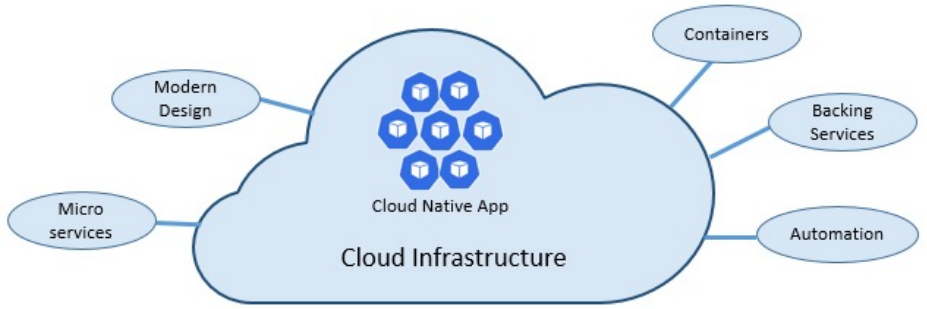


UI/UX Design Patterns

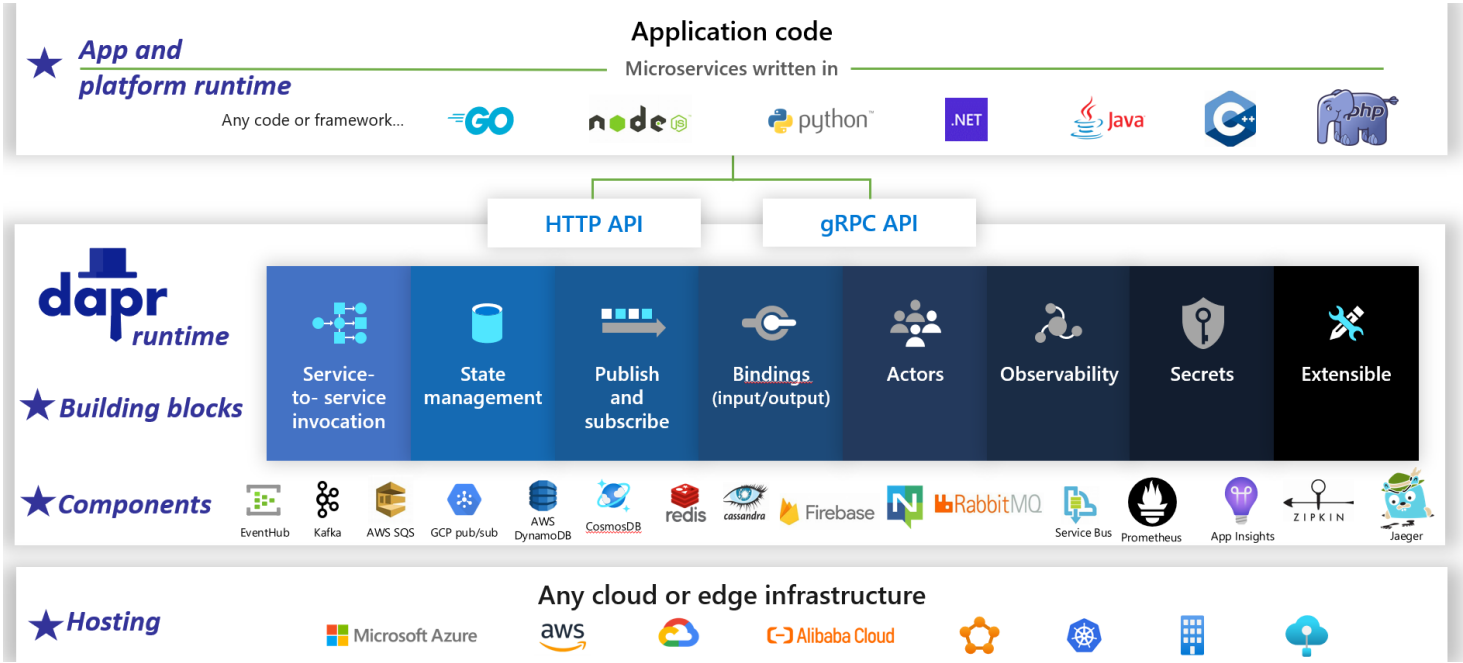
Check your website here



Application Design Patterns

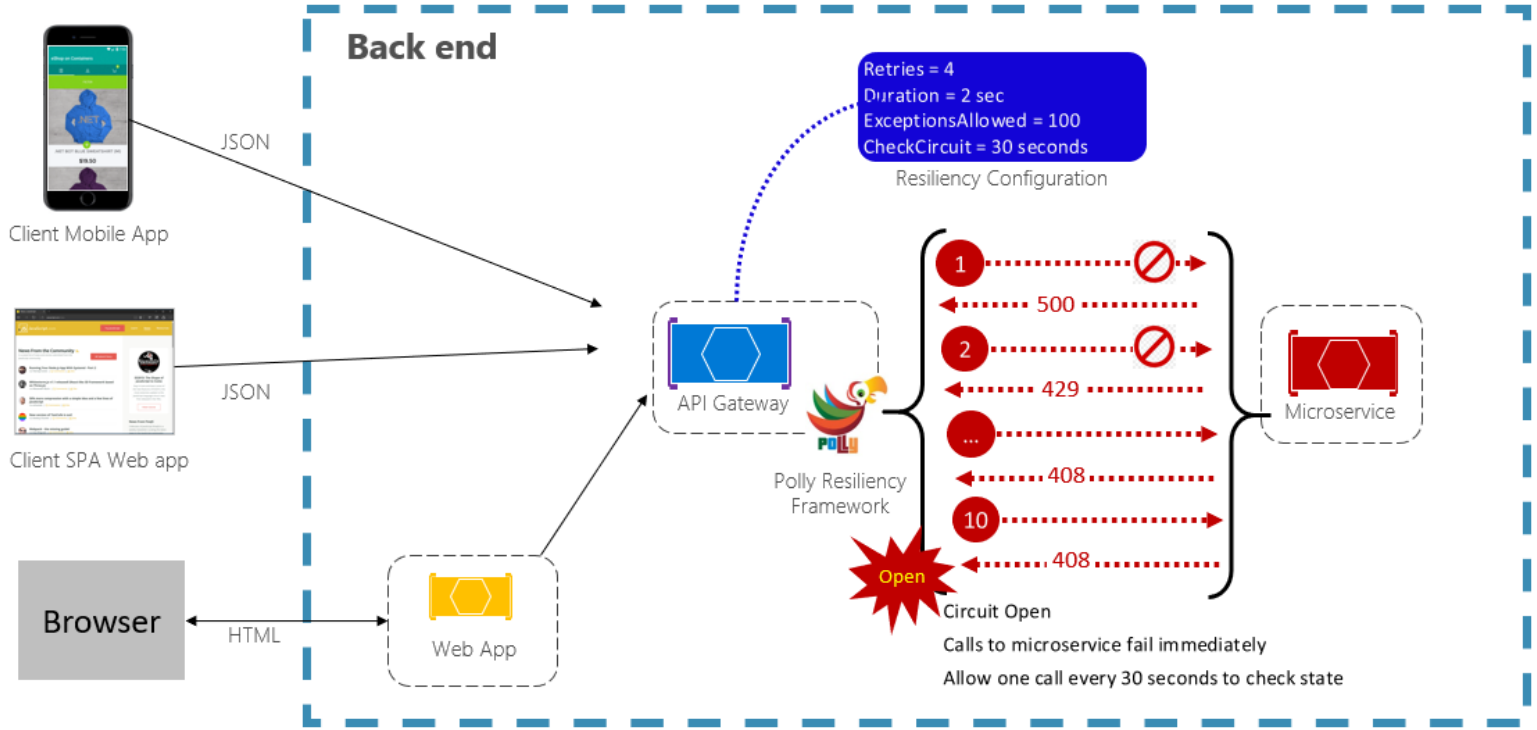


- Leverage **cloud-native** design patterns
- Evaluate moving monoliths to a microservice architecture
- Containerize workloads where applicable



Application Design
Patterns

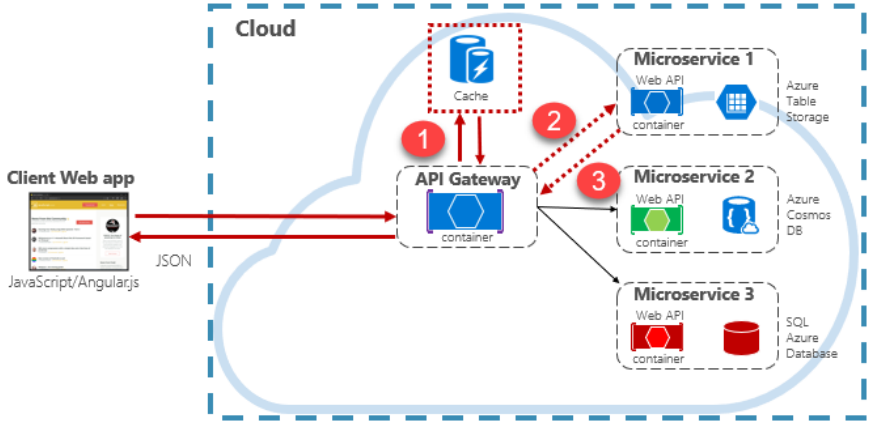
- Consider using **Circuit Breaker** patterns



Application Design Patterns

Improved API efficiency through:

- Advanced request throttling
- Cache responses
- selective queries
- encoding



REST API

```
REST REQUEST
GET https://sample.com/person/1
```

```
REST JSON
{
  “firstName”: “John”,
  “middleName”: “Andrew”
  “lastName”: “Smith”,
  “email”: “jas1992@gmail.com”,
  “relationshipStatus”: “single”
}
```

VS

GraphQL

```
GRAPHQL QUERY
{
person {
      firstName
      lastName
   }
}
```

```
GRAPHQL JSON
{
  “data”: {
    “person”: {
      “firstName”: “John”,
      “lastName”: “Smith”,
    }
  }
}
```

Application Design
Patterns

- Ensure backward software compatibility
- Reduces the number of mandatory updates
- Make performance improvements updates mandatory

Coding Patterns

- Consider using a more **energy efficient programming language**
- Check out the [whitepaper](#) from 2017

	Energy
(c) C	1.00
(c) Rust	1.03
(c) C++	1.34
(c) Ada	1.70
(v) Java	1.98
(c) Pascal	2.14
(c) Chapel	2.18
(v) Lisp	2.27
(c) Ocaml	2.40
(c) Fortran	2.52
(c) Swift	2.79
(c) Haskell	3.10
(v) C#	3.14
(c) Go	3.23
(i) Dart	3.83
(v) F#	4.13
(i) JavaScript	4.45
(v) Racket	7.91
(i) TypeScript	21.50
(i) Hack	24.02
(i) PHP	29.30
(v) Erlang	42.23
(i) Lua	45.98
(i) Jruby	46.54
(i) Ruby	69.91
(i) Python	75.88
(i) Perl	79.58

Energy

	Time
(c) C	1.00
(c) Rust	1.04
(c) C++	1.56
(c) Ada	1.85
(v) Java	1.89
(c) Chapel	2.14
(c) Go	2.83
(c) Pascal	3.02
(c) Ocaml	3.09
(v) C#	3.14
(v) Lisp	3.40
(c) Haskell	3.55
(c) Swift	4.20
(c) Fortran	4.20
(v) F#	6.30
(i) JavaScript	6.52
(i) Dart	6.67
(v) Racket	11.27
(i) Hack	26.99
(i) PHP	27.64
(v) Erlang	36.71
(i) Jruby	43.44
(i) TypeScript	46.20
(i) Ruby	59.34
(i) Perl	65.79
(i) Python	71.90
(i) Lua	82.91

Execution Time

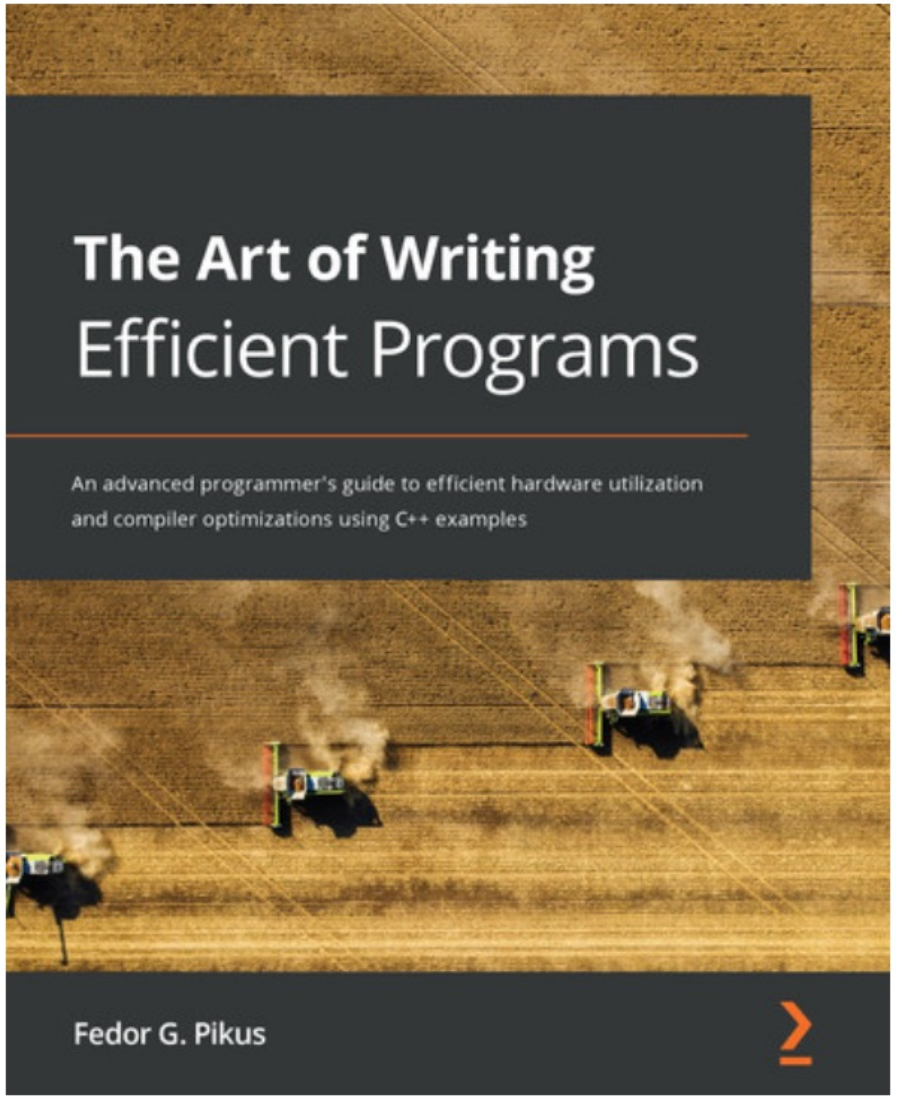
	Mb
(c) Pascal	1.00
(c) Go	1.05
(c) C	1.17
(c) Fortran	1.24
(c) C++	1.34
(c) Ada	1.47
(c) Rust	1.54
(v) Lisp	1.92
(c) Haskell	2.45
(i) PHP	2.57
(c) Swift	2.71
(i) Python	2.80
(c) Ocaml	2.82
(v) C#	2.85
(i) Hack	3.34
(v) Racket	3.52
(i) Ruby	3.97
(c) Chapel	4.00
(v) F#	4.25
(i) JavaScript	4.59
(i) TypeScript	4.69
(v) Java	6.01
(i) Perl	6.62
(i) Lua	6.72
(v) Erlang	7.20
(i) Dart	8.64
(i) Jruby	19.84

Memory

Coding Patterns

High code density & efficient runtime behavior

- Choosing the right algorithm
- Efficient use of CPU resources
- Efficient use of memory
- Avoiding unnecessary computations
- Efficient use of concurrency and multithreading
- Effective use of programming language features
- Measuring performance



Coding Patterns

- **Optimize for async access patterns**

```
    int total = _urlList.Select(url => ProcessUrl(url)).Sum();

    stopwatch.Stop();
    _resultsTextBox.Text += $"\nTotal bytes returned:  {total:#,#}";
    _resultsTextBox.Text += $"\nElapsed time:          {stopwatch.Elapsed}\n";
}

1 reference
private int ProcessUrl(string url)
{
    using var memoryStream = new MemoryStream();
    var webReq = (HttpWebRequest)WebRequest.Create(url);   // sync
```

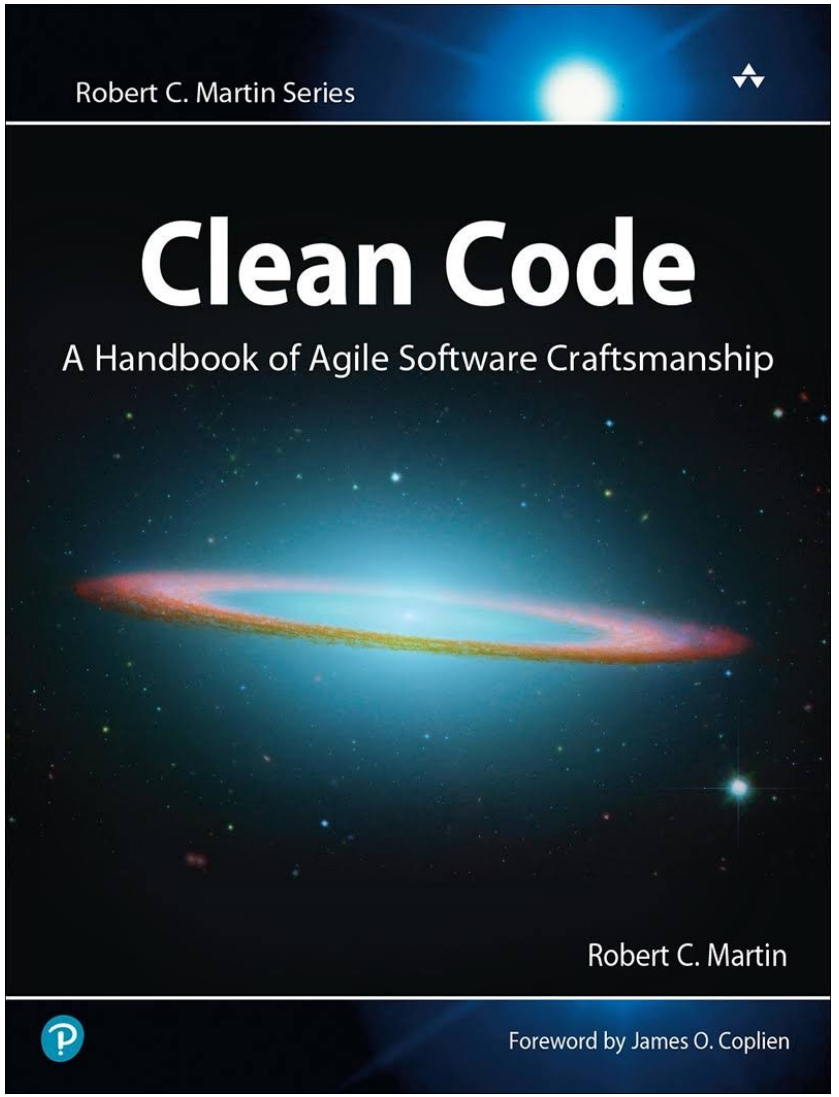
	Type of Program	Total data transfer	Latency (secs)	Energy	Unit of energy	MWHr	kWHr	mtCo2 Values	
1	Sync	1.4 MB	10	61854	mJoules	1.73191E-08	1.73191E-05	0.016470483	mtCo2eq
2	Serial Async	1.4 MB	15	39855	mJoules	1.11594E-08	1.11594E-05	0.010612589	mtCo2eq
3	Parallel Async	1.4 MB	0.99	17798	mJoules	4.98344E-09	4.98344E-06	0.004739251	mtCo2eq

Coding Patterns

Avoiding "dead code"

- Keep an eye on code coverage
- Pay attention to clean code principles:

Reduce complexity
Increase comprehensibility
Prevent overconfigurability
Use of Dependency Injection
Prefer polymorphism over if/else
Use descriptive and unique names
Writing small functions that do one thing at a time
Use clear and short comments
Do not comment out code but delete it
Implementing short independent modules – reusability



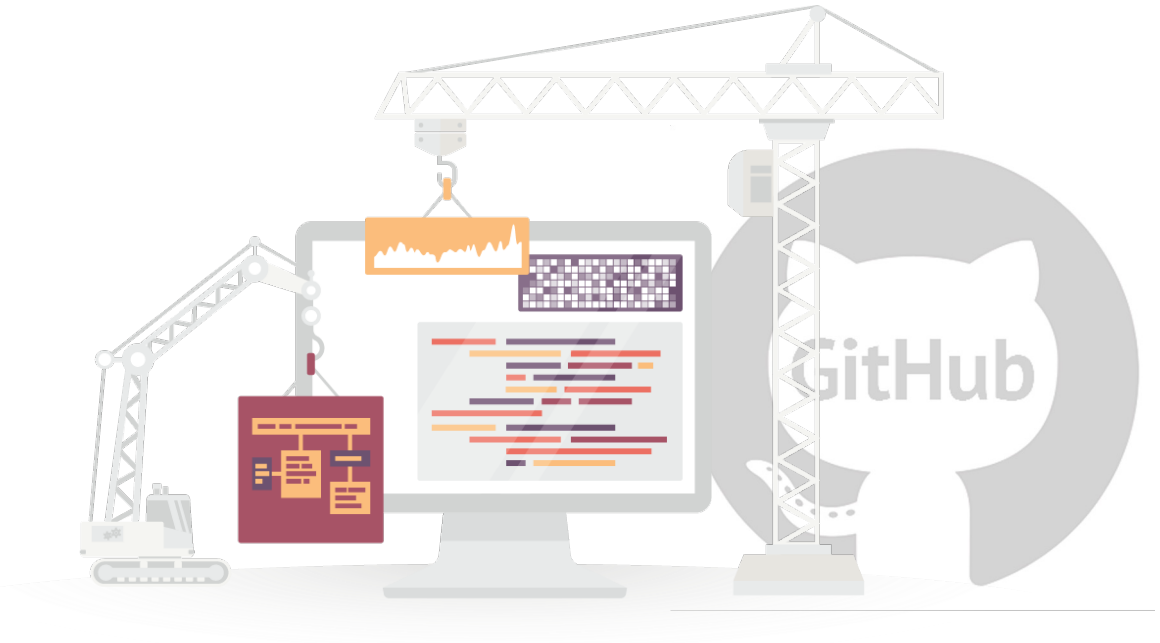
Coding
Patterns

- Assess where parallelization is possible
- Consider using chaos engineering practices
- Establish CPU and Memory thresholds in testing

Azure Chaos Studio

Coding
Patterns

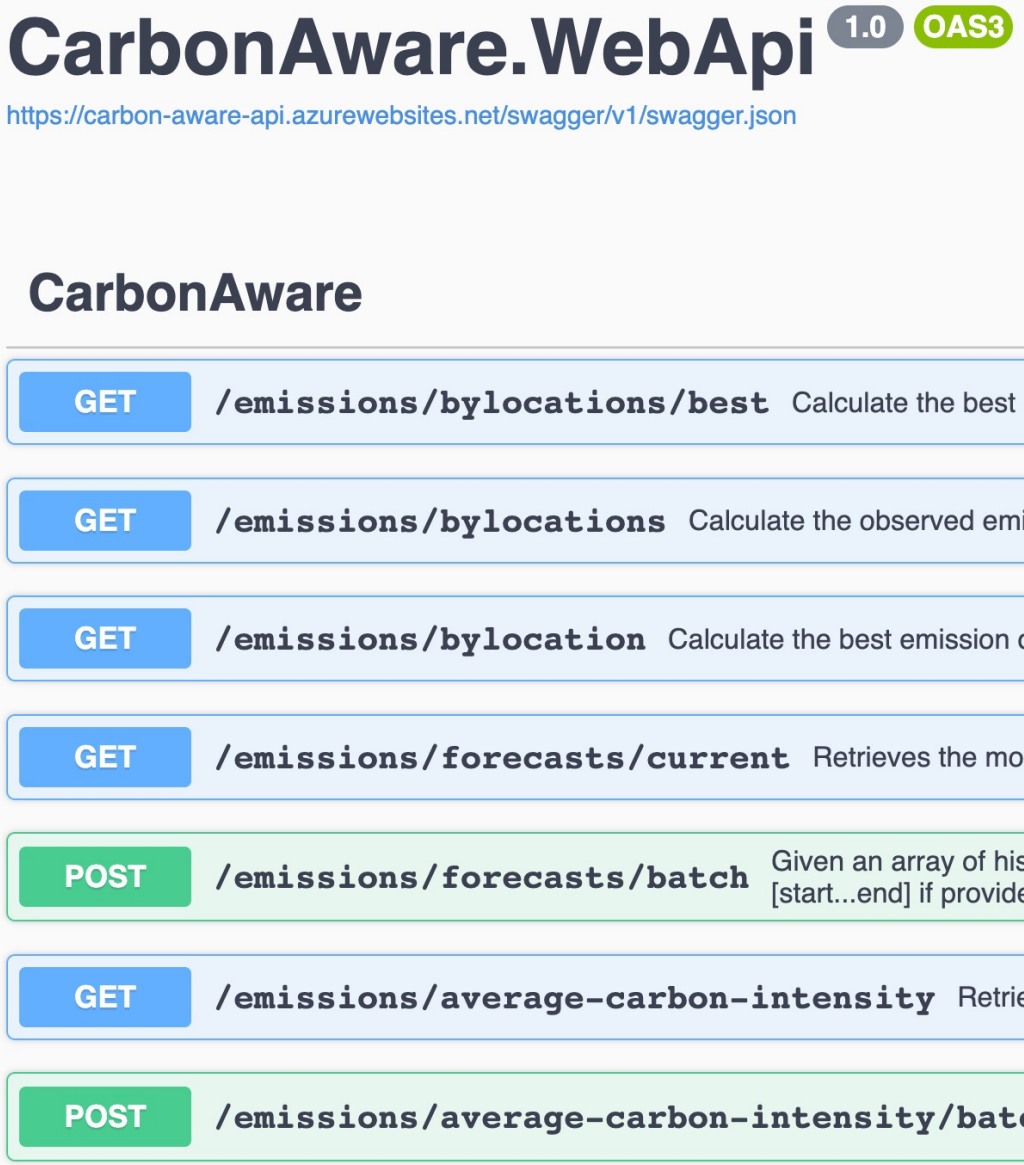
- Open source
- Modular code
- Monitoring dashboards
- Architectures
- Processes



Platform Patterns

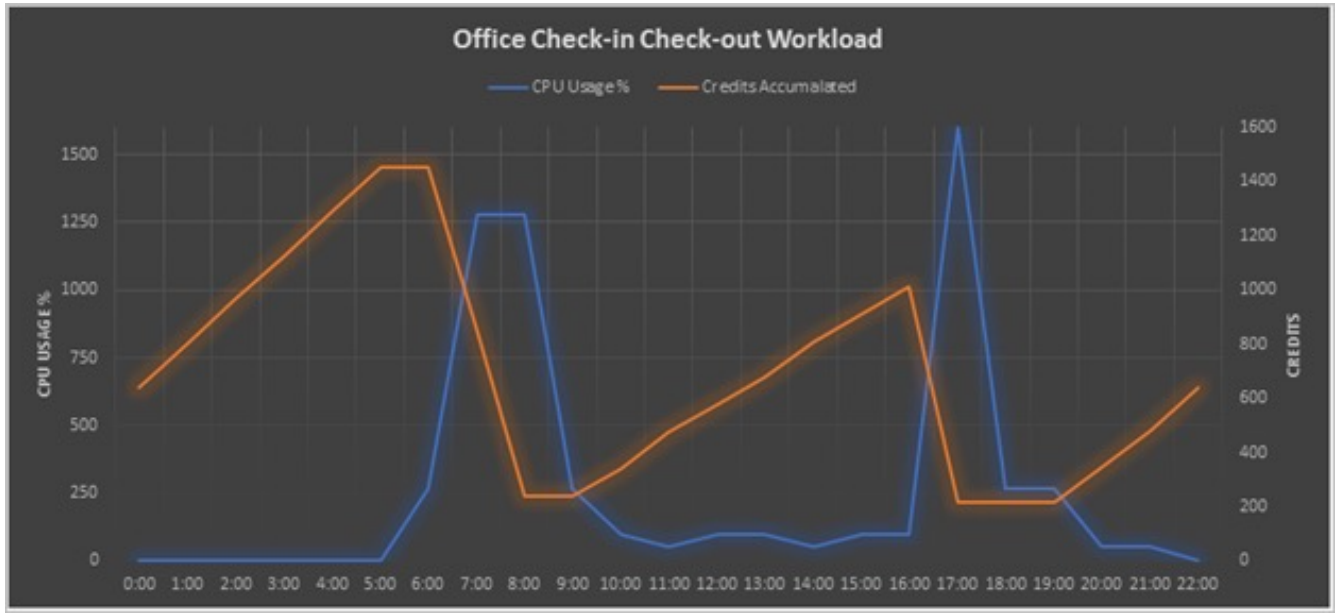
- Deploy to low-carbon regions
- Process when the carbon intensity is low
- Choose datacenters close to the customer
- Run batch workloads and testing during low-carbon intensity periods

→ Consider using the **Carbon-aware SDK**



Platform
Patterns

- Evaluate moving to **PaaS** and **serverless** workloads
- **Turn off** workloads **outside of business** hours
- Utilize **auto-scaling** and **bursting** capabilities

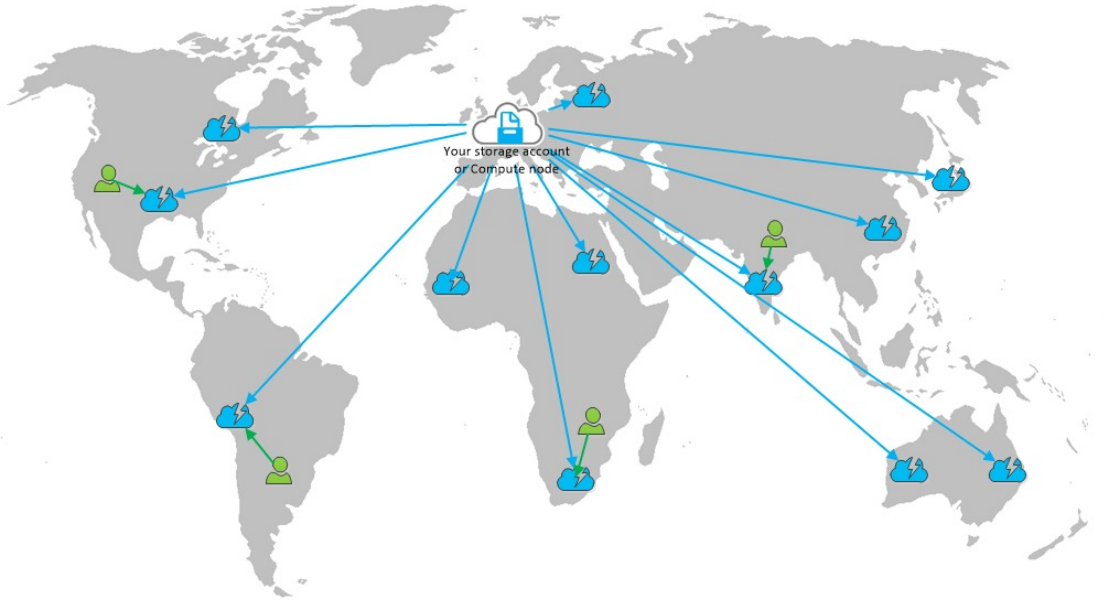


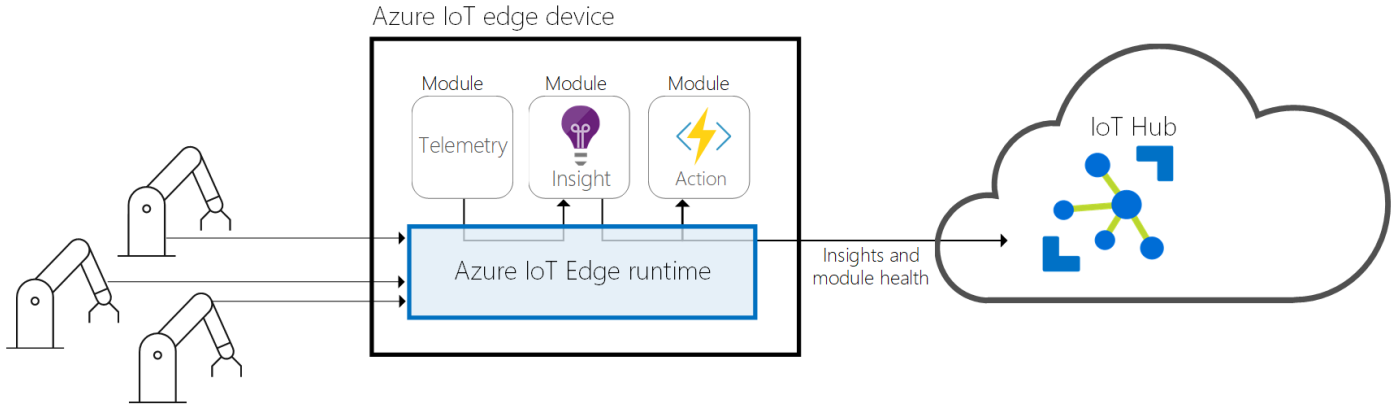
Platform
Patterns

- Match the scalability needs
- Delete zombie workloads
- Consider using Arm-based processors for Virtual Machines

Platform
Network Patterns

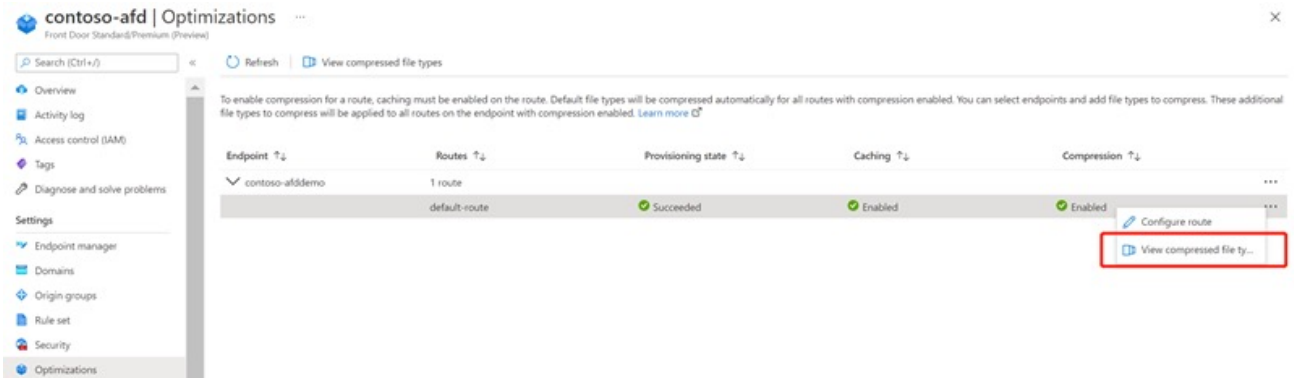
- Consider using CDN
- Computing on the edge
- Enable network file compression
- Avoid sending data to other regions or over the internet - if possible





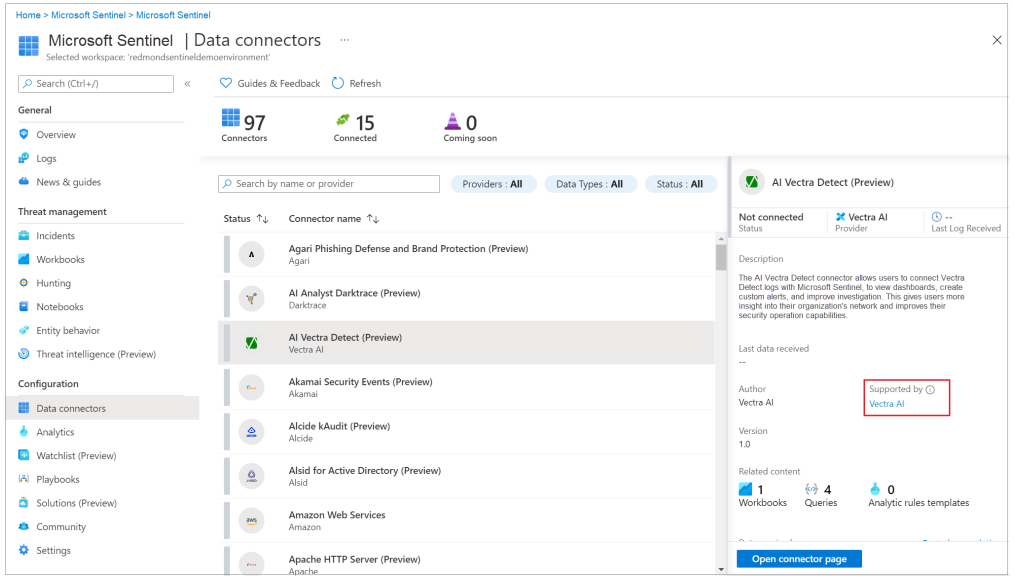
Platform
Data Patterns

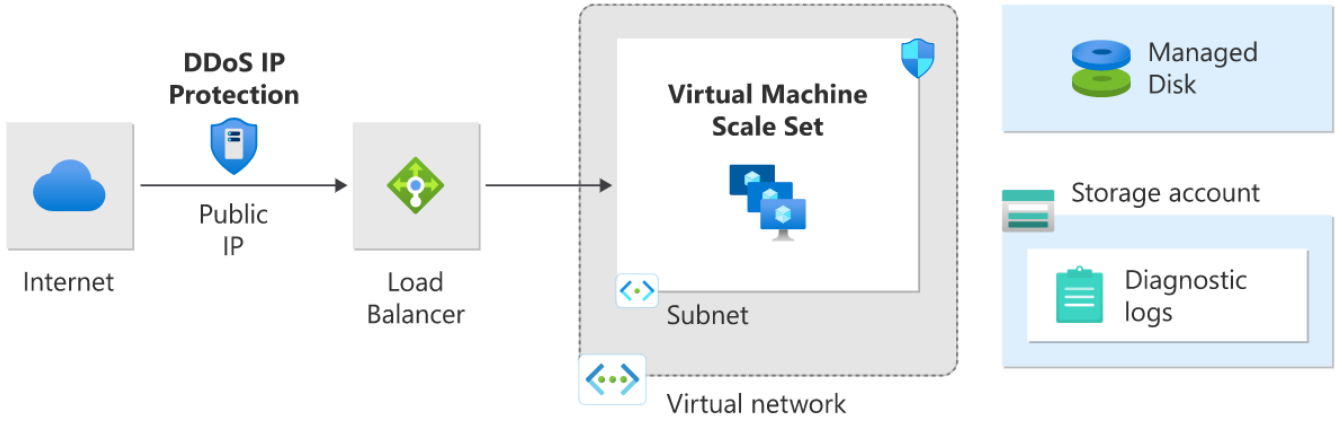
- Use storage compression
- Optimize database query performance
- Use the best suited storage access tier
- Design your backup strategy wisely
- Revisit log data storage strategy



Platform
Security Patterns

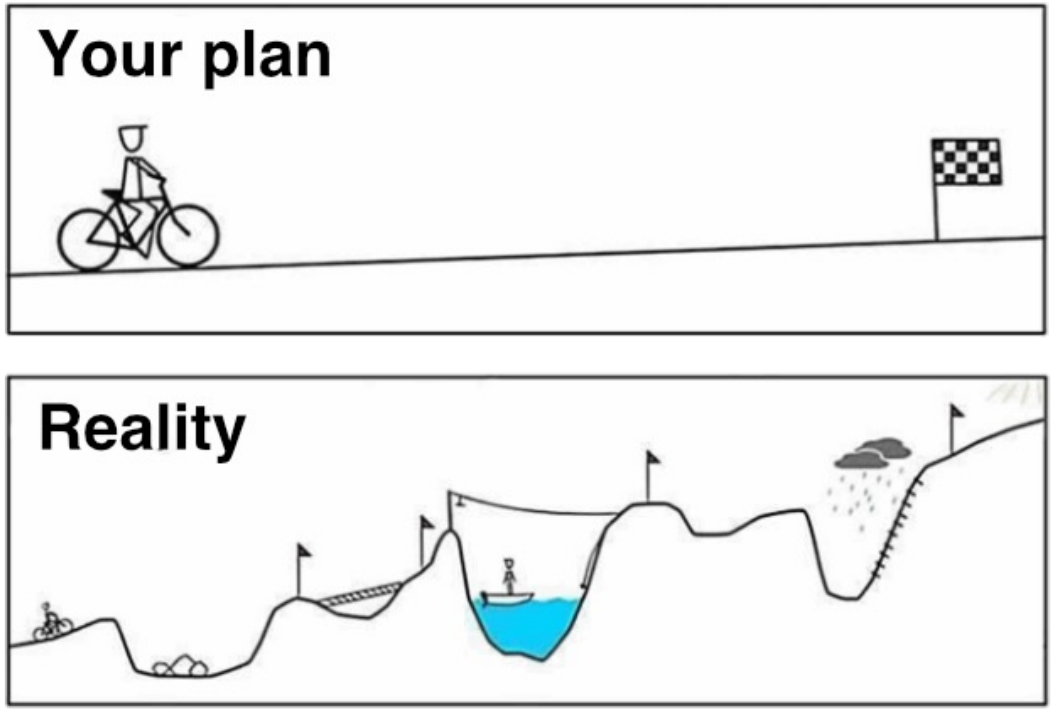
- Use DDoS protection
- Evaluate whether to use TLS termination
- Use network security tools with auto-scaling
- Minimize routing from endpoints to the dest
- Avoid centralized routing- and firewall design
- Use cloud native log collection methods





Culture
Executional Patterns

- Work on what matters most
- Small increments
- Regular reality checks



Culture

Operational Patterns

- Camera on/off
- Consider using file shares or hosting services

Culture

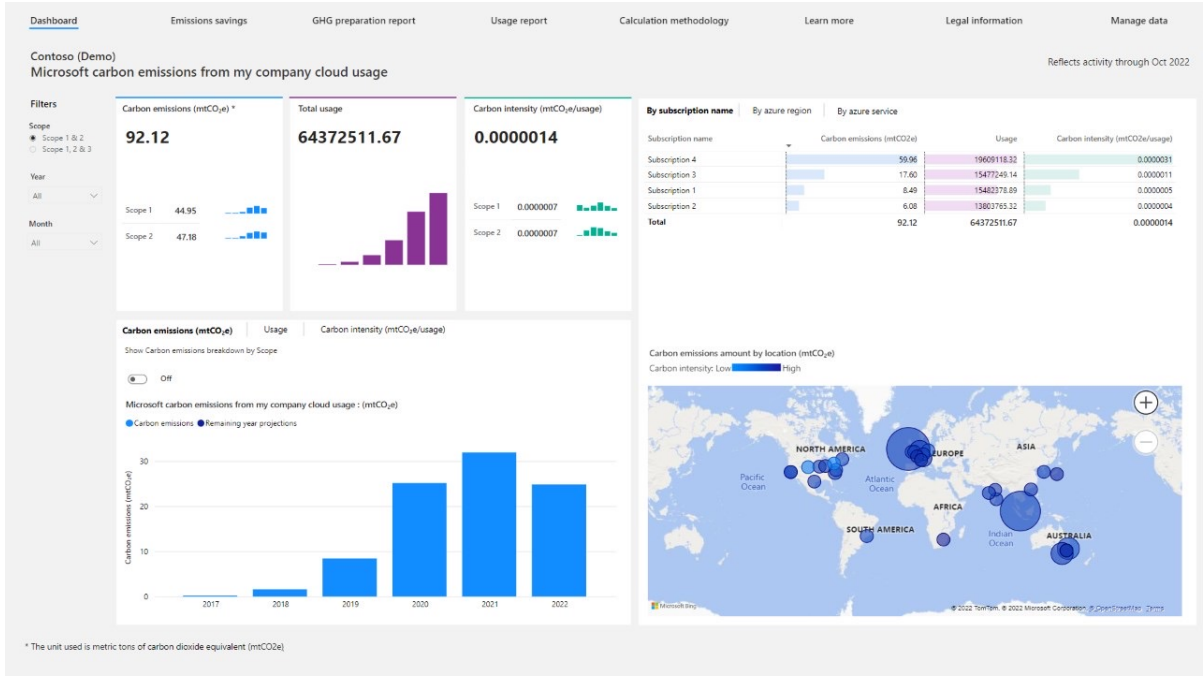
Operational Patterns

- Consider buying refurbished devices
- Less frequent hardware renewal
- Hibernate or simply shut it down!

Culture

Tracking Patterns

- Use the **Emissions Impact Dashboard**
- Define emissions **targets**
- **Identify** metrics & set **goals**
- Use **cost** optimization as a **proxy**



Culture

Community Patterns

- Create a sustainability **community**
- Leverage **internal trainings** sessions
- **Share best practices** across teams
- Plan for **incentives**

Resources

https://greensoftware.foundation
https://github.com/Green-Software-Foundation/carbon-aware-sdk
https://patterns.greensoftware.foundation/
https://www.watttime.org/
https://learn.microsoft.com/en-us/azure/architecture/framework/sustainability/
https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE4RwfV
https://www.microsoft.com/de-de/sustainability/learning-center?rtc=1
https://learn.microsoft.com/en-us/training/modules/sustainable-software-engineering-overview/
https://infrastructuremap.microsoft.com/fact-sheets
https://www.websitecarbon.com/
https://www.amazon.com/-/de/dp/B09BZTGJM2/ref=tmm_kin_swatch_0?_encoding=UTF8&qid=&sr=
https://www.amazon.com/-/de/dp/B001GSTOAM/ref=tmm_kin_swatch_0?_encoding=UTF8&qid=1666864762&sr=1-1


aka.ms/azdevhub
Azure Developer Community Hub
Your one stop shop for developer resources on Azure


Microsoft
Thank you