101 GREEN SOFTWARE

A PRACTICAL GUIDE FOR DEVELOPERS & ARCHITECTS

IOANNIS KOLAXIS

101 Green Software

A Practical Guide for Developers & Architects

Ioannis Kolaxis

This book is for sale at http://leanpub.com/green-software

This version was published on 2023-10-21

ISBN 978-618-00-3480-6

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Tweet This Book!

Please help Ioannis Kolaxis by spreading the word about this book on Twitter!

The suggested tweet for this book is:

As software engineers, we can all help prevent climate change! I just started my journey by reading the #GreenSoftwareBook, learning how to build green software

The suggested hashtag for this book is #GreenSoftwareBook.

Find out what other people are saying about the book by clicking on this link to search for this hashtag on Twitter:

#GreenSoftwareBook

Dedicated to the three generations of teachers who taught me how to read, write and teach.
Dedicated to my wife and daughters who inspire me for a better tomorrow.
All together they contributed to the very existence of this book.

Contents

About this book

It's the year 2050.

We did nothing to stop climate change.

In the North and South Poles, the ice has melted completely. The rise of sea level flooded all the coastal cities around the world. Billions of people are homeless.

The weather conditions are extreme, severely affecting food production, which cannot meet worldwide demand.

Is that the future we want to live?

What can you do to prevent climate change?

Human activities, such as burning fuels, produce carbon dioxide (CO_2) and other gases known as greenhouse gases. These gases are present in the earth's atmosphere and trap heat. They let the sunlight pass through the atmosphere, but they prevent the heat that the sunlight brings from leaving the atmosphere. You can think of them as a blanket wrapped around the earth.

To prevent climate change, we need to reduce the emissions of greenhouse gases.

What can you do as a software engineer to help prevent climate change?

I used to think that recycling our garbage and driving electric cars was the best we could do! However, it turns out that we can do a lot more to address global warming by contributing to reducing greenhouse gas emissions.

Where do these emissions come from? As shown below, electricity and heat producers are the largest contributor to global carbon emissions [IEA19]:

- Electricity & Heat Producers: **41,84%**
- Transport: 24,45%
- Industry: 18,60%
- Residential: 5,85%

You can help reduce the carbon emissions in these sectors by working in one of these directions:

- Building software that enables businesses to reduce their emissions, also known as **software decarbonization**

- Building software that itself has reduced or zero emissions, also known as **decarbonized software**

We usually refer to greenhouse gas emissions as **carbon emissions** since carbon dioxide (CO_2) is the primary greenhouse gas emitted through human activities. The term **decarbonization** refers to the process of reducing carbon emissions.

Software decarbonization

As software engineers, we can build software applications that transform the way traditional businesses work, essentially inventing new ways of doing things that help reduce carbon emissions.

For example, think about how video-conferencing software (like Zoom and Microsoft Teams) transformed the traditional workplace by enabling people to work from home. Thanks to this software we don't have to commute by car to the office, thus contributing to a significant decrease in greenhouse gases emitted by vehicles.

The topic of digitally transforming existing businesses to reduce their carbon emissions is not explored in this book.

Decarbonized software

As software engineers, we build applications that run on servers. These servers are hosted in data centers, and they consume electricity. Application users rely on a network, most commonly the internet, to access our applications and exchange data.

According to the following data [IEA20], data centers and data transmission networks consume approximately **2%** of the global electricity:

- **Data centers** consumed approximately 1% of global electricity,
- **Data transmission networks** consumed 1,1-1,4% of global electricity.

Software engineers can develop software applications that need fewer resources (CPU, RAM, data stored, data transferred over the network), consuming less electricity and thus emitting less carbon.

Following the example of video-conferencing software (like Zoom), we would be working to make these applications more energy efficient. For instance, we could use audio and video codecs that consume less CPU and less network bandwidth.

This book explains how to enhance software applications to reduce or eliminate their carbon emissions. This topic is known as green or decarbonized software.

How can you build green software?

As a software engineer, you can help prevent climate change by developing green software that has reduced or zero carbon emissions.

This book helps you learn how to build green software, enabling you to apply gained knowledge into your daily work directly. It provides practical guidelines that you can follow to:

- re-architect existing software applications or
- build new ones that have a reduced environmental footprint.

The author's perspective

I have been building software applications for 17+ years, working in different roles, from developer to architect. As an architect, I have been reducing software application complexity and cost, with the latter measured by a decrease in the cloud bills.

When running your applications on the cloud, you pay according to the resources (CPU, RAM, data stored & transferred over the network) that your applications consume. The more resources your applications consume, the higher your cloud bill is.

Although my goal has been to improve software applications' cost efficiency, this was accomplished by re-architecting them to consume fewer resources, thus enhancing their resource efficiency as a by-product. Improved resource efficiency leads to higher energy efficiency since fewer resources consume less electricity.

Green software engineering is an emerging field. However, we can draw on our experience of building lightweight, cost-efficient software to make our existing applications greener. In this book, I share my professional experience combined with the latest practices on building green software. It's a book written by a developer for developers!

My vision is to educate every developer on building green software, hoping that each one of us can contribute to reducing carbon emissions. When all these individual efforts add up, they can make a huge difference, having a measurable impact on saving our planet.

I intentionally wrote this book to be as concise as possible. Time is of the essence; we need to quickly onboard green software professionals to help reverse global warming.

If you are involved in building software applications and aspire to help prevent climate change, read on! This book will empower you to fight climate change by adopting green software engineering practices.

Join the discussion on our private group to provide your feedback about this book and share your experience building green software: https://groups.google.com/g/green-software-book

1. Don’t send your data around the world

Data transmission networks consume more electricity than data centers.

Data transmission networks connect computers, allowing them to exchange information.

When you send your data around the world, you contribute to global warming.

In this chapter, you will learn practices that reduce the electricity consumption of data networks.

When you travel by car, you use Google Maps or a similar application to navigate you to your destination. This application usually proposes alternative routes, prompting you to choose:

- The fastest route, which usually involves taking a highway, or
- The route with the shortest distance

Nowadays, Google Maps allows you to choose the most fuel-efficient route. [GoogleMaps21]

It calculates this route based on factors like traffic congestion and road incline. Since your vehicle will consume the least fuel, it will emit less carbon, thus having a minimal impact on the environment [1].

When you transfer data over the network, do you ever look at the route taken by your data to reach their destination?

1.1 Place your servers close to your end-users

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

1.2 Reduce the size of data sent over the network

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

[1] That’s an excellent example of software enabling decarbonization in the sector of transportation, which is the second-largest contributor to global carbon emissions.

Filter out unnecessary data

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Compress data

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Cache data

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

1.3 Process data at its source

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

1.4 Adopt energy-efficient protocols

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

How HTTP/2 works

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Energy efficiency of HTTP/2

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

1.5 Summary

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

2. Run your software on carbon-free electricity

Don't run your code after midnight!

Do you have any batch jobs running late at night?

If yes, these are likely consuming electricity generated by coal power plants, which emit the most carbon compared to all other sources of electricity.

In this chapter, you will learn about the different sources of electricity and understand how demand for electricity shapes the mix of supplied energy. This knowledge will enable you to run your code on the cleanest available electricity with the lowest carbon emissions.

2.1 Carbon-free electricity explained

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

2.2 Electricity mix in power grids

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Base load

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Intermediate load

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Peak load

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

2.3 Assessing the electricity mix

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

2.4 Choose a data center that consumes carbon-free energy

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

2.5 Run your code when the sun shines

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

2.6 Send your code where the sun shines

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Run your code where it's cold outside

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Build a green Kubernetes scheduler

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

2.7 Summary

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

3. Get rid of redundant servers

50% of the servers in your data center don't do anything. They waste energy!

Half of the servers in your data center are redundant.

These servers are idle most of the time, waiting for a failure to happen so they can take over and handle requests.

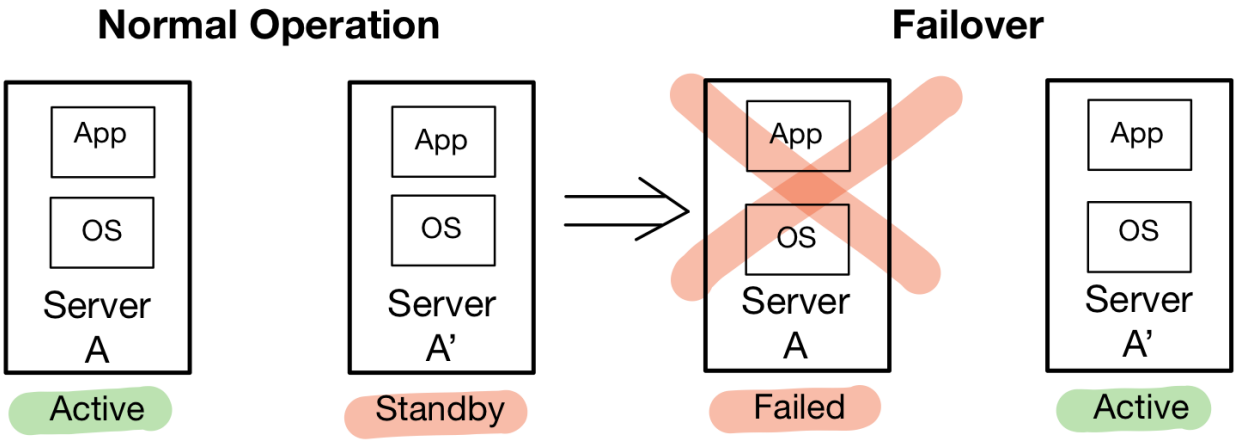
While they wait, they waste energy and inflate your cloud bills!

In this chapter, you will learn how to eliminate these redundant servers, reducing your application's energy consumption and your cloud bills by up to 50%.

3.1 Redundant servers explained

When failures happen to our applications, for example, a hardware fault in a server, we don't want our end-users to notice this failure but rather wish to continue doing business as usual.

For this reason, we usually have one or more redundant servers in "warm standby" mode, doing nothing other than waiting for a failure to occur to an active server so that they can take over. The following diagram illustrates this: upon the failure of Server A, we perform a failover to the previously idle Server A', which assumes responsibility for fulfilling existing and new user requests.



High-Availability Using a Redundant/Warm Standby Server

An enterprise software application typically consists of several servers, with up to 50% being redundant, wasting resources while waiting for a failure to happen.

Citroën 2CV

Citroën 2CV Sahara (produced from 1958 until 1966) is a car that has two identical engines for redundancy.

The first engine is in the front, and the second is in the rear of the vehicle.

It has two separate petrol tanks, each supplying fuel to a different engine.

This car was designed for the desert Sahara, where a failure in the engine could be fatal.

If you plan to go to Sahara, choose a car with two engines for redundancy.

But when you go to the cloud, your software applications don't need redundant servers!

Can you think of an alternative, high-availability model for your software applications without redundant servers?

3.2 Adopt containers for high availability

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

3.3 Leverage automation for high availability

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

3.4 Get rid of redundant data centers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

3.5 Trade-offs

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

3.6 Summary

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

4. Introduce a stop/start mechanism

Idle applications should not consume energy

Modern cars have a stop-start system[1] that automatically stops the engine when you are not moving due to a traffic jam or while waiting in front of a red traffic light.

The stop-start system reduces fuel consumption by shutting down the car engine whenever it's not needed, thus reducing carbon emissions.

Older cars did not have this system, which meant they would still emit carbon while their engines were idle.

Can you think of a similar case for software applications?

Some applications may be idle most of the time, waiting for an event to happen, so they can start processing something related to this event. While these applications wait, they consume resources (CPU, RAM) and electricity.

This chapter explains how to re-architect your applications to eliminate their energy consumption while idle.

4.1 Run your application as serverless functions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

[1] https://en.wikipedia.org/wiki/Start-stop_system

4.2 Run your application as serverless containers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

4.3 Prefer serverless services

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

4.4 Trade-offs

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

4.5 Summary

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

5. Adopt green programming languages

What kind of "fuel" does your software run on?

Today cars run on a diverse range of fuels: diesel, petrol, ethanol, propane, compressed natural gas, electricity, and hydrogen.

Some cars are friendlier to the environment, depending on their fuel.

Electric cars are becoming mainstream, having the lowest environmental impact compared to fossil fuel cars.

What happens with software applications?

Is there some "fuel" that software applications run on?

We write software applications in different programming languages. Some languages are greener, consuming less energy.

In this chapter, you will learn about the environmental impact of each programming language, helping you choose the greenest languages to rewrite your existing applications or develop new ones.

5.1 Prefer compiled languages

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

5.2 Should you rewrite your application?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

5.3 Summary

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

6. Decarbonize Java applications

You can make your Java applications greener!

Duke, the Java mascot

In 1996, Java 1.0, the first version of Java, was released.

Since then, Java has been one of the most popular programming languages. It has a thriving developer community and the wealthiest ecosystem of tools, frameworks, and libraries.

Many software applications worldwide run on Java, such as Wikipedia Search and Minecraft. [1]

As seen in the previous chapter, Java ranks well in energy efficiency. However, you can make your Java applications greener by applying the actions presented in this chapter.

Given the global footprint of Java applications, your actions can have a significant impact in helping prevent climate change.

6.1 Upgrade to the latest version of Java

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

[1] https://blogs.oracle.com/javamagazine/post/the-top-25-greatest-java-apps-ever-written

Benefit from compact strings

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Enable Application Class-Data Sharing

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Shrink your Java Runtime Environment

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

6.2 Use a JVM that consumes less memory

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

6.3 Use a Java framework that consumes less memory

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

6.4 Summary

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

7. How green is your software?

If you can't measure it, you can't improve it.

New electric appliances have an energy label[1] (A, B, C, etc.), helping us buy the most energy-efficient one. For example, energy labels indicate the annual power consumption in kWh for refrigerators; those consuming the least power get the highest rating.

Can we assign energy labels to software applications?

This chapter explains how to assess the energy efficiency of software applications. You will learn how to measure the environmental footprint of your software, observing how this footprint changes while applying the actions proposed throughout this book.

7.1 Reduce cloud costs for greener applications

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Get your cloud application Well-Architected

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

[1] https://en.wikipedia.org/wiki/European_Union_energy_label

7.2 Can you assign energy labels to software applications?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

What are the boundaries of your application?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

How does your application compare to others?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

7.3 Measure software carbon emissions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

From annual reporting to daily monitoring of carbon emissions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

How to monitor software carbon emissions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Identify carbon emission hotspots

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

Track proxy sustainability metrics

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

How will you monitor the greenness of your software?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

7.4 Summary

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

8. How will you contribute?

A journey of a thousand miles begins with a single step.

As software engineers, we can contribute to preventing climate change by eliminating the carbon emissions of our software applications.

We cannot do this on our own. We are going to need your help!

What actions will you take to start building greener software?

8.1 Set priorities

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

8.2 Take your first steps

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

8.3 Track your progress

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

8.4 Reinvent software development

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/green-software.

References

[Barroso07] L. A. Barroso and U. Hölzle, "The Case for Energy-Proportional Computing," in Computer, vol. 40, no. 12, pp. 33-37, Dec. 2007, doi: 10.1109/MC.2007.443.
https://www.cs.princeton.edu/courses/archive/spring13/cos598C/Barroso07_EnergyProp-clean.pdf

[Chowdhury16] S. A. Chowdhury, V. Sapra and A. Hindle, "Client-Side Energy Efficiency of HTTP/2 for Web and Mobile App Developers," 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER), 2016, pp. 529-540, doi: 10.1109/SANER.2016.77.
https://softwareprocess.es/pubs/chowdhury2016SANER-http2.pdf

[EIA20] "About 25% of U.S. power plants can start up within an hour", U.S. Energy Information Administration, Nov 2020.
https://www.eia.gov/todayinenergy/detail.php?id=45956

[GoogleMaps21] "Redefining what a map can be with new information and AI", Mar 2021.
https://blog.google/products/maps/redefining-what-map-can-be-new-information-and-ai/

[Hohpe19] Hohpe Gregor *Cloud Strategy - A Decision-Based Approach to Successful Cloud Migration.* 2019. ISBN:9798665253046

[Howarth20] Howarth Jesse, "Why Discord is switching from Go to Rust", Feb 2020.
https://discord.com/blog/why-discord-is-switching-from-go-to-rust

[IEA19] Based on IEA data from the IEA (2019) Greenhouse Gas Emissions from Energy, CO2 emissions by sector, https://www.iea.org/data-and-statistics/data-browser/?country=WORLD&fuel=CO2%20emissions&indicator=CO2BySector. All rights reserved; as modified by Ioannis Kolaxis.

[IEA20] Based on IEA data from the IEA (2020) Data Centers and Data Transmission Networks, https://www.iea.org/reports/data-centres-and-data-transmission-networks. All rights reserved.

[James19] James Aled and Daniel Schien. "A Low Carbon Kubernetes Scheduler." ICT4S (2019).
http://ceur-ws.org/Vol-2382/ICT4S2019_paper_28.pdf

[JetBrains21] "The State of Developer Ecosystem 2021", Jul 2021.
https://www.jetbrains.com/lp/devecosystem-2021/

[Koningstein21] Koningstein Ross, "We now do more computing where there's cleaner energy", Google, May 2021.
https://blog.google/outreach-initiatives/sustainability/carbon-aware-computing-location/

[Korando19] Korando Billy, "OpenJ9 class sharing in Docker containers", IBM Developer, Feb 2019.
https://developer.ibm.com/articles/eclipse-openj9-class-sharing-in-docker-containers

[Moomaw11] W. Moomaw, P. Burgherr, G. Heath, M. Lenzen, J. Nyboer, A. Verbruggen, 2011: Annex II: Methodology. In IPCC Special Report on Renewable Energy Sources and Climate Change

Mitigation [O. Edenhofer, R. Pichs-Madruga, Y. Sokona, K. Seyboth, P. Matschoss, S. Kadner, T. Zwickel, P. Eickemeier, G. Hansen, S. Schlömer, C. von Stechow (eds)], Cambridge University Press, Cambridge, United Kingdom and New York, NY, USA.
http://www.ipcc-wg3.de/report/IPCC_SRREN_Annex_II.pdf

[Nygard2018] Nygard Michael T. *Release It! Second Edition.* The Pragmatic Programmers, 2018. ISBN:9781680502398

[Pereira17] R. Pereira, M. Couto, F. Ribeiro, R. Rua, J. Cunha, J. Fernandes, J. Saraiva, "Energy efficiency across programming languages: How do energy, time, and memory relate?", International Conference on Software Language Engineering, Oct 2017.
https://sites.google.com/view/energy-efficiency-languages/home

[Philippot17] Philippot Olivier, "HTTP2, Latency and Energy", Greenspector, Mar 2017.
https://greenspector.com/en/http2-latency-and-energy/

[Radovanovic21] Radovanovic Ana, Ross Koningstein, Ian Schneider, Bokan Chen, Alexandre Nobrega Duarte, Binz Roy, Diyue Xiao, Maya Haridasan, Patrick Hung, Nick Care, Saurav Talukdar, E. Mullen, Kendal Smith, MariEllen Cottman and Walfredo Cirne. "Carbon-Aware Computing for Datacenters." ArXiv abs/2106.11750 (2021)
https://arxiv.org/pdf/2106.11750.pdf

[Rais19] Issam Raïs, Daniel Balouek-Thomert, Anne-Cécile Orgerie, Laurent Lefèvre, Manish Parashar. Leveraging energy-efficient non-lossy compression for data-intensive applications. HPCS 2019 - 17th International Conference on High Performance Computing & Simulation, Jul 2019, Dublin, Ireland. pp.1-7. hal-02179621
https://hal.archives-ouvertes.fr/hal-02179621

[Rocher20] Rocher Graeme, "Micronaut vs Quarkus vs Spring Boot Performance on JDK 14", Apr 2020.
https://micronaut.io/2020/04/07/micronaut-vs-quarkus-vs-spring-boot-performance-on-jdk-14/

[Smeets19] Smeets Maarten, "Microservice framework startup time on different JVMs (AOT and JIT)", Sep 2019.
https://technology.amis.nl/software-development/java/microservice-framework-startup-time-on-different-jvms-aot-and-jit/

[SpeedTest21] "Global Network Speeds", Oct 2021.
https://www.speedtest.net/global-index

[Zhang14] H. Zhang, S. Shao, H. Xu, H. Zou, and C. Tian, "Free cooling of data centers: A review," Renewable and Sustainable Energy Reviews, vol. 35, pp. 171 – 182, 2014.
http://www.sciencedirect.com/science/article/pii/S1364032114002445

Author biography

Ioannis Kolaxis

Ioannis Kolaxis is a Director at Accenture Technology Sustainability Innovation, focusing on Green Software innovations. He has been previously developing software for Atos, IBM, and Siemens.

Ioannis was a kid in primary school when he first saw the green screen with the blinking cursor of an Amstrad CPC6128. This computer opened a new world to him, a world of continuous exploration and creation through programming.

He invented a novel way of working remotely, having filed five patents for software solutions[1]. He was awarded twice the first prize for coming up with the most innovative ideas at Atos Innovation Week 2020 and 2021.

Ioannis frequently speaks at conferences and meetups, including GOTO Amsterdam 2022, Devoxx Ukraine 2021, and Oracle Code One 2019.

Follow him on Twitter and LinkedIn and join the conversation on how to build green software by posting your comments:
https://twitter.com/IoannisKolaxis
https://www.linkedin.com/in/ioannis-kolaxis/

Visit his website to learn more: https://kolaxis.dev

[1] https://patents.google.com/?inventor=Ioannis+Kolaxis