

A new approach to coding: resource-efficient programming

09/11/2023 Information technology is the basis for innovations in a wide range of industries. At the same time, however, IT energy consumption could lead to major challenges in the future. That's why computer scientists aim to use green coding to significantly reduce the CO_2 emissions of IT.

Whether it's the internet, data centers or artificial intelligence: Wherever intensive computing is carried out, energy consumption and the associated greenhouse gas emissions are high. According to a report by the Association for Computing Machinery (ACM), information and communication technologies (ICT) and global air traffic are now roughly on a par in terms of their climate impact: Depending on the study quoted, the ICT industry accounts for 1.8 to 2.8 percent of annual greenhouse gas emissions. If the effects of hardware production are also factored in, its share is almost four percent.

Air traffic is estimated to account for around 2.5 percent. ICT energy demand is likely to continue to increase significantly in the future: According to the ACM, if current trends continue, it could be responsible for one-third of all global greenhouse gas emissions in 2050. "Computing-intensive processes such as Big Data, the Internet of Things, and artificial intelligence are, in some cases, very

energy-hungry," as Prof. Volker Wohlgemuth of the Berlin University of Applied Sciences reports. "While they can make a major contribution to sustainable development, they must themselves be developed in the most resource-efficient way possible."

More energy-efficient hardware and a climate-neutral power supply are likely to lead to improvements in this regard. Yet software development can also make a noticeable contribution—through green coding. Green coding is a software development approach that aims to reduce the resources and energy required to design, create, process and publish a software project. It has been an issue in vehicle development for some time: ECUs exhibit extreme limitations in terms of memory and computing power, and therefore need to be programmed very efficiently. But even in this case, it could be possible to save energy—by intelligently distributing the computing operations: energy-saving ones in the vehicle and more computationally intensive ones in the cloud.

Problems with bloatware

There are studies that attest how great the impact of software is on the energy consumption and greenhouse gas emissions of information technologies. For example, a study commissioned by the German Federal Environment Agency shows how differently two word processing programs utilize the resources of a computer: In a standard usage scenario, one software consumed 3.6 watt-hours of energy, while a competing product came in at just 0.93 watt-hours. "Although both programs perform the same tasks, program two requires only about a quarter of electrical energy and is therefore significantly more energy efficient," the authors of the study found. A comparison of three internet browsers revealed a similar picture: Under comparable conditions, the energy consumption during use was 1.95, 0.91 and 0.66 watt-hours, respectively.

When idle, their processor utilization actually fluctuated between 0.8 and 12 percent. One reason for the poor values is bloatware, this being software that is encumbered with a multitude of—often little used—functions and which is therefore not very energy efficient. The utilized programming languages also result in significant differences: C, for example, which was developed in the early 1970s, and the relatively new language Rust perform best in terms of energy consumption, while the widely used languages Ruby and Python result in significant increases in power consumption. In a comparative test, they required 70 and 76 times more power than C, respectively, for the same tasks—not least because both are recompiled (interpreted) during each program runtime, whereas with C or Rust this process occurs only once before the software is started (compilation).

"However, some programming languages offer much better support through specific libraries for a problem that may not exist in others," Wohlgemuth points out. "So you always have to look at the individual case." To minimize the impact of software development on the environment, green coding applies a range of approaches. "These include software architecture, implementation, methodology and platforms," Wohlgemuth says. "There is a multitude of different ways to program power-saving mechanisms." For example, if you don't write a mathematical function (calculating the factorial of a natural number) yourself in Python, but instead use the predefined variant from a function library

written in C, you can reduce the energy consumption by over 90 percent. A similar reduction was also measured for the calculation of random numbers.

Avoiding unnecessary code

There is also potential for optimization when it comes to the use of open source software libraries: Many contain code that is never used and leads to unnecessary energy consumption during compilation. Savings can also be achieved by keeping network traffic to a minimum in distributed software applications. “This can be achieved, for example, by using fewer high-resolution images or by using binary rather than text-based file formats,” Wohlgemuth said. “In addition, running computations as locally as possible instead of deep in the cloud can be worthwhile—but there is still a need for research in this area.” Software also has implications for the service life of IT hardware. “If resource requirements keep increasing, for example due to bloatware, computers have to be replaced at an unnecessarily early stage, which leads to additional consumption of energy and raw materials,” Wohlgemuth says.

“Programmers should also think about this when they write software.” To take account of these and various other requirements for the sustainability of code, and to recognize particularly green products, the German Federal Ministry for the Environment has extended its Blue Angel eco-label to software. “This covers various aspects of sustainability,” explains Joseph De Veaugh-Geiss, who works on the Blue Angel for Free and Open Source Software project for the software community KDE.

“In addition to energy consumption, it's about the service life of the hardware, but also the autonomy of the user. Among the criteria is the ability to remain free of unwanted advertising—which in the EU alone produces as much greenhouse gas emissions annually as a city the size of Turin.”

Growing interest

So far, only one piece of software has been awarded the Blue Angel. “Politicians should use the Blue Angel as an award criterion for public contracts,” says Wohlgemuth, who has the impression that green coding is only slowly gaining traction among businesses. In research and teaching, on the other hand, the topic is already receiving much more attention, he says, because computer science has also recognized that programs and hardware can have a major impact on the environment and the climate—in both a positive and negative sense. “Green coding can be applied in any industry or used as a criterion for awarding contracts, and there is particularly great potential in areas such as IT, finance, automotive or online retail,” Wohlgemuth says. “That's why it's important for us to create awareness among developers about the immense leverage they have for more climate protection in their work.”

Info

Text first published in the Porsche Engineering Magazine, issue 2/2023

Text: Christian Buck

Copyright: All images, videos and audio files published in this article are subject to copyright. Reproduction in whole or in part is not permitted without the written consent of Dr. Ing. h.c. F. Porsche AG. Please contact newsroom@porsche.com for further information.

MEDIA ENQUIRIES



Sandro Kälin

Head of Communications Porsche Schweiz AG
+41 41 487 91 16
sandro.kaelin@porsche.ch

Link Collection

Link to this article
https://newsroom.porsche.com/fr_CH/2023/innovation/porsche-engineering-green-coding-34340.html

Media Package
https://pmdb.porsche.de/newsroomzips/fe3c282f-c4ca-42b5-b3ba-0d97bf2fceec.zip