KEYNOTE

Software durability (is software engineering sustainable?)

SÉBASTIEN RUMLEY, University of Applied Sciences Western Switzerland, Delemont, JU, Switzerland

Published: 28 May 2025

Citation in BibTeX format

CF '25 Companion: 22nd ACM International Conference on Computing Frontiers
May 28 - 30, 2025
Cagliari, Italy

Conference Sponsors:
SIGMICRO

https://doi.org/10.1145/3706594.3734583
ISBN: 9798400713934

Software durability (is software engineering sustainable?)

Sébastien Rumley
University of Applied Sciences and Arts Western Switzerland (HES-SO)
Fribourg, Switzerland
HEIA-FR
Institute of Artificial Intelligence and Complex Systems (iCoSys)
Fribourg, Switzerland
sebastien.rumley@hes-so.ch

Abstract

The scientific and engineering communities are increasingly recognizing the alarming lack of sustainability in many human practices. This realization is triggering an unexpected paradigm shift in engineering. Traditionally, engineers are driven by the desire to invent, design, and build. However, these actions often deplete natural resources, generate waste, and emit greenhouse gases (GHGs). To move toward sustainability, engineers must focus less on building new things and more on reusing existing ones. This shift requires a growing emphasis on maintenance and, by extension, *durability* - the ability of something to remain functional without requiring excessive maintenance or repair.

In the IT sector, sustainability is often associated with the manufacturing of computers and the energy required to operate them. However, as IT engineers and scientists, we do more than just invent, design and build hardware (and the power infrastructure that supports it). Most of us deal with software - we invent, design and write code. Is that unsustainable – at least as long as the software is not executed? Is software development itself a polluting activity?

Well, yes, it is. For starters, writing code requires a computer, whose production depletes natural resources, generates waste, and emits GHGs. Moreover, most of our codes and many coding practices are not durable. Dijkstra wrote in 1988 that if we wish to count lines of code, we should not regard them as "lines produced" but as "lines spent" [1]. The real cost of software still lies in maintaining it - and perhaps even more than before. A famous cartoon from Randall Munroe accurately depicts how modern software sometimes relies on a myriad of packages, including some unexpected ones [2]. And many have warned about the possible consequences of fragile software [3]. Despite this, we continue to generate more and more code, especially now with IA capable of writing it for us.

This talk exhibits the link between software durability and sustainability, and explores practical ways to make software engineering more sustainable and durable.

ACM Reference Format:
Sébastien Rumley. 2025. Software durability (is software engineering sustainable?). In *22nd ACM International Conference on Computing Frontiers (CF Companion '25), May 28–30, 2025, Cagliari, Italy*. ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3706594.3734583

CF Companion '25, Cagliari, Italy

ACM ISBN 979-8-4007-1393-4/25/05
https://doi.org/10.1145/3706594.3734583

Biography

Sébastien Rumley is an Associate Professor at the University of Applied Sciences and Arts Western Switzerland (HES-SO), affiliated with the iCoSys institute. Prof. Rumley earned his M.S and PhD from EPFL in 2005 and 2011, respectively. Following his academic achievements, he joined the Lightwave Research Laboratory at Columbia University, New York, where he remained until 2018. After gaining valuable industry experience, Prof. Rumley assumed the role of Associate Professor of Software Engineering in 2021. His extensive research background encompasses optics, optical networking, and computer architecture. He is widely recognized as an expert in high-performance optical interconnects. Prof. Rumley's current areas of interest include computing infrastructure (both cloud and physical), energy efficiency and sustainability in computing and IA, and integration of IA within SMEs. With over 80 publications to his name, he has received multiple awards such as the Best Paper Award at the SuperComputing conference in 2016 and the Hans Meurer Award (Best Paper award of the ISC-HPC conference) in 2018.

References

[1] E. W. Dijkstra. 1988-12. On the cruelty of really teaching computing science. https://www.cs.utexas.edu/~EWD/ewd10xx/EWD1036.PDF
[2] Randall Munroe et al. [n. d.]. https://www.explainxkcd.com/wiki/index.php/2347:_Dependency
[3] James Somers. 2017-9. The Coming Software Apocalypse. *The Atlantic* (2017-9). https://www.theatlantic.com/technology/archive/2017/09/saving-the-world-from-code/540393/