Shaping the hidden environmental impacts of software

GSHA 2023

Thibault Simon, Pierre Rust, Romain Rouvoy
Software quickly became central

Which led to increased complexity

Pong, 1972
Which led to increased complexity

35 millions lines of code\textsuperscript{1}

\textsuperscript{1}https://chromium.googlesource.com/chromium/src
Hardware is complex

developers use different abstraction levels

- Physics
- Binary
- Machine code
- Assembly code
- C Language
- Python
Software scale grew *a lot*

2.99 billion monthly active users\(^1\)

---

*Visualizing Friendships*, Facebook engineering, 2010

---

\(^1\) First Quarter 2023 Financial Highlights, Meta
Software scale grew *a lot*

Even more abstraction layers were added
Which distribute responsibility (separation of concerns)
Which **distribute** responsibility (*separation of concerns*) and **dilute** the underlying infrastructure's environmental footprint
While ICT's environmental footprint comes from hardware, responsibility lies within software.
Energy consumption is driven by software

Between 4 and 8% of global electricity use\textsuperscript{1, 2}


Device manufacturing is driven by software

Mineral extraction has contributed to environmental degradation, population displacement, violent conflicts and human rights violations\(^1\)

\[^1\] Mapping Mining to the SDGs: An Atlas, UN, 2016
Software can drive e-waste generation

Up to 20% of device renewal in EU are caused by software\(^1\)

Credit: Muntaka Chasant

[1] Obsolescence logicielle, Rapport CGEDD, 2020
Can I, as a software engineer, have a direct effect on the repairability of hardware devices? On the tracing of e-waste? On the social consequences of mining?
Can I, as a software engineer, have a direct effect on the repairability of hardware devices? On the tracing of e-waste? On the social consequences of mining?
Assigning responsibility to others does not trigger actions

Adapted from *The Triangle of Inaction*, Pierre Peyretou
What is my responsibility as a software engineer?
What is my responsibility as a software engineer?

to act *at my level*
The environmental impact levels of ICT\textsuperscript{1,2}:

1. **Direct** effects: negative environmental impacts of ICT hardware life cycle
2. **Indirect** or enabling effects: optimization, obsolescence...
3. **Systemic** effects: long-term reaction (notably rebound effects)


Indirect effects depend on external factors out of developer's control

Developers should focus on lowering their software's direct impact.
Developers should focus on lowering their software's direct impact by lowering the pressure they place on underlying layers.
Avoid falling into an *inaction loop*
Monitor and reduce resources *used* and *reserved*
Challenge the need for abstraction layers
Virtual resources from any abstraction layer can be linked to hardware
Which can be translated to an environmental impact over multiple categories through:

- Energy consumed
- Hardware embodied emissions
Which can be translated to an environmental impact over multiple categories through:

- Energy consumed
- Hardware embodied emissions

To measure, understand and lower it
Conclusion

While software is virtual, its environmental footprint is not
Conclusion

While software is virtual, its environmental footprint is not.

If you wait for underlying components to reduce their footprint, they will do the same.
Conclusion

While software is virtual, its environmental footprint is not.

If you wait for underlying components to reduce their footprint, they will do the same.

We need to go further than energy consumption and co2 emissions.
Future work

Holistic approach on software life cycle¹

Future work

Holistic approach on software life cycle\textsuperscript{1}

Lowering resources waste at all layers

\textsuperscript{1} Uncovering the Environmental Impact of Software Life Cycle. Preprint: https://hal.science/hal-04082263/
Thank you for your attention

Any questions?