Kitware Software Sustainability Matrix

Date: 7 August 2024
Presented by: Bill Hoffman (Kitware, Inc.) and Will Schroeder (Kitware, Inc.)

(The slides are available via the link in the page's sidebar.)

Q: Does software need to be “sustainable” in order for software t o be “sustained”? I.e., are there examples of software projects that have been sustained for decades with lots of users that would score very low on objective
software quality/sustainability metrics? If so, what is the real definition of “software sustainability”?

A: Part of the measurement of sustainability includes monitoring the health and vitality of an associated software community. A large and energetic community can make up for many deficiencies related to software sustainability. One of the underlying assumptions worth mentioning is that the software is open: without it, there is little likelihood that software can be sustained into the future.

Q: How do you measure if a piece of software written in the past has been sustained?

A: Several ways: Is it still available for use, does it still run on common computing platforms.

Q: Have you done a scientific study that correlates your metrics to quality outcomes? Have you done any studies that show cause and effect (that improving a metric improves quality)?

A: No. This is a long-term goal. Currently, many organizations such as the Linux Foundation, government/research agencies, open source communities, and companies like Kitware are engaged in conversations about software sustainability, with the intention of finding ways to ensure that critical computing software remains available into the future.

Q: Can you please comment on how OSS commercialization and business models can help improve the sustainability of software. Many research softwares are difficult to maintain past the discrete, one time budgets allocated by a grant, so what are some options for financially sustaining software after these grants run out?

A: (Participant comment from Dan Katz) ReSA keeps a list of funding opportunities for new software and for software maintenance: https://www.researchsoft.org/funding-opportunities/ (which anyone can add to)

Kitware's sustainability matrix lists a "Business Model" metric. This is a reflection that resources are necessary to sustain software over the long run. In some cases, depending on scale, an energetic community is enough; however, for many research softwares, funding agencies and commercial/not-for-profit benefactors can and do play a critical role. Also, companies like Kitware use a hybrid business model that combines funding from multiple sources / customers, combined with a vision for the software that has been used to grow and support systems like VTK for decades.

Q: Does Kitware have tools which are adopting ML? If so, any concerns about reproducibility?

Kitware is involved in building and using ML tools. Yes, there are significant concerns about reproducibility. When testing conventional software, there are typically clearly known outputs given a set of input. However, even in such software, algorithms based on random seeding etc, or parallel computing where threads may run in unexpected order, must be tested carefully. ML takes this to a whole new level and it is not clear exactly how to test this, given output is a function of training sets, as well as the ML model itself.

Q: Software that ran in DOS does not run on any OS anymore. It is just an example of software that is not sustainable. Is sustainability really an absolute measure of the worth of software?

A: Sustainability is a measure of availability of software over time, it does not measure the worth of software. However, there appears to be a correlation between long-term sustainability and impact/value of software.High impact/value translates into supportive communities and organizations, which have a vested interest in making sure software remains available.

Q: Since you don't have any scientific studies that provide insight into the value of these metrics, it seems there is room for improvement. Yes, having testing vs not having testing is clearly better, but that's like saying a person can survive better by drinking fluids vs not. That's not a compelling story.

A: The question assumes that people actually test software in a formalized, rigorous way. In our experience this is far from true, especially in a research setting where publishing the next paper, or producing the next demo, are the primary drivers. We routinely see research teams who only sporadically test software, or only test small parts of it when producing results for publication. Part of the intention of devising these metrics is to identify and encourage practices that improve the health of software. We are in the early days of recognizing the problem (of software health and sustainability) with future research efforts expected to formalize approaches. To return to the example mentioned by the questioner, there are plenty of folks who are dehydrated, overhydrated, or drink too much suboptimal fluid (e.g., beer) - we want to more specifically guide organizations and community towards best practices that ensure software improves and can be used into the future.

Q: Is there a relation between sustainable software and sustainable data (formats)?

A: Most definitely. Reproducibility in science, and software / data interoperability benefits greatly when standard, sustainable data formats are used. Typically sustainable data formats are paired with software modules that demonstrate how to access (read / write / modify) data. Basically, anything that can be done to reduce friction, and increase the value of software relative to the effort to use it, tends to encourage the growth of communities and other resources consistent with ensuring that software remains useful into the future – i.e., is sustainable.