Software Development in a Context of Sustainability

Alexander Lanin[1]

[1]Affiliation not available

April 28, 2020

Software Development in a Context of Sustainability

Introduction

Software development is one of the greatest achievements of a modern technology and it offers an opportunity for individuals and corporations, as well as governments, to conduct and execute tasks on a high efficiency level unimaginable in the past. Moreover, from one perspective, the software serves as technology's significant step towards sustainability. In essence, the creation of software poses hardly any threat to the physical environment. For instance, it is possible to transfer an application from one hardware device to the other. It is further possible to dispose of the application if it fails (Kern et al., 2018). Such a dimension represents the value of software in reducing wastes in the physical world. However, most studies and arguments still cite the sustainability issue of information technology (IT), which primarily applies software in the running of applications and processes. It is, therefore, prudent to understand the dynamics of sustainability without necessarily basing on the business outcomes of software development. The paper seeks to capture the problems of software sustainability before stating probable goals.

Software Sustainability

The sustainability of businesses and processes ought to incorporate three principal dimensions. Before the term sustainability, pioneering studies addressed the triple bottom line. It refers to the factors that corporations need to address in the effort of developing lasting businesses. Such aspects include the social justice of the company, its effect on the environment as well as the income it generates for the organization (Savitz and Weber, 2006). Therefore, a sustainable business is that addresses the people, environment, and business. Such an organization is built to last and can be resilient to most forces in the operating surrounding. The triple bottom line relates to the stakeholders whose interests business authorities need to take into account. The stakeholders include the ecosystem and the social and financial facets of operations. In the effort to realize sustainability, business entities ought to determine ways of working that not only amass profits for the companies but safeguard the social and environmental interests (Beer, 2009). Such a move is crucial to building organizational resilience.

Sustainability, in the context of the United Nations, involves working towards a globe that works in such a manner that minimizes the devastating effects on humanity and the environment. The 17 sustainable development goals of the UN point to the need for looking past the profits. Such a move is crucial for not only organizational resilience but for limiting the adverse effect of resource exploitation on people and the ecological system (UN, 2017). The UN sustainable development goals (SDGs) indicate the body's efforts to

balance between people, profits, and the environment. They aim at reducing poverty through the creation of opportunities. They further advocate for good health among people, which can only be achieved by preserving the environment and limiting the adverse impact of technological advancement.

When viewed in the context of the triple bottom line, it is evident that software development presents an additional barrier to sustainability. Most of the applications that corporations use in IT do not last long. They ought to advance, and the same extends to organizations, who update their software periodically. Changing the applications by deleting the previous ones and replacing them with the recent updates hardly impacts the immediate environment directly. Indirectly, the update of software versions implies that their creation seeks to match the latest hardware devices (Kern et al., 2018). Subsequently, the old hardware items applied in IT become incompatible with the recently updated software. From such a lens, the indirect effect of software development is evident. It is an indication that while software creation barely affects the ecology, the effect results from the hardware gadgets that deploy the applications.

The primary sustainability challenge of software development, therefore, is the effect it has on resource and power consumption (Mahmoud and Ahmad, 2013). Sustainability goes beyond developing products designed to meet long-term business goals. Instead, it encompasses addressing the societal, economic, and ecological impacts of processes that seek to satisfy human needs. For that reason, new aspirations in software development have aimed at reducing the use of power and resources because of application deployment. The problem of software creation happens in the hardware devices from which the application is run. Hence, gadgets consume more power and resources than the ecological system can comfortably manage. The development of green software purposes to curb the consumption of energy and resources in running IT software. It further aims at reducing the number of hardware wastes that result from obsolete gadgets. Most importantly, it endeavors to address the emissions from the devices, in terms of heat and CO2.

Another problem with the deployment of IT software is the impact it has on the social aspect of humanity. Social sustainability continues to be the greatest challenge of creating computer software (Al Hinai and Chitchyan, 2014). Such an issue occurs at a time when most human activities are dependent on application use. The implementation spreads from offering government services, to communication and other political aspects like voting. Unfortunately, humanity has dedicated little attention and resources to addressing the social effects of software development. Software sustainability has aimed to reduce the resource and energy consumption of the hardware devices using the applications. Yet, the social sustainability of advancement has hardly captured any attention and dedication. There are several social sustainability indicators, including employment, health, security, education, equity, and resilience, among others. Most software reports and studies show that governments and organizations have hardly dedicated any efforts and resources to the social effect of computer applications.

Most of the measures towards the sustainability of IT software have focused on the ecological and financial concerns of corporations. For instance, developers have created applications that can help in environmental conservation measures. Such software aids in tracking gas emissions, predicting disasters and climatic shifts, and the management of energy use and CO2 emission (Oyedeji, Seffah, and Penzenstadler, 2018). The green software movement, on the other hand, has sought to minimize energy consumption and resource use. It has concentrated on the life cycle of the software, mostly aimed at reducing the rate at which computer hardware gadgets become incompatible with the subsequent application updates. The creation of the GREENSOFT model comes in the effort to support application developers in the production, maintenance, and usage of computer software. Notably, it exhibits the failure of the model to address the social sustainability concerns of software development.

Another dimension to software sustainability focuses on the effects that arise from the creation, usage, and maintenance of computer applications. From such a perspective, it is evident that the sustainability of software can be examined from the impact it has on computer hardware. Therefore, the advancement seeks to incorporate the effect of the update and creation of applications in the development of sustainable software (Albertao et al., 2010). The sustainability performance metrics, for example, analyze the sustainability of an application at the end of its cycle before proposing the improvement measures. Viewing software

sustainability addresses not only the energy and resource consumption rates of the applications but the social effect of their usage. Hence, the development of sustainable software comes with the possession of features that seek to minimize the negative impact of the creations on the stakeholders of the companies. According to the triple bottom line approach of sustainability, such stakeholders include the people, profits, and environment, which are the primary factors.

While current models of software sustainability aim at reducing the detrimental effects of computer application usage on the environment, little attention has addressed the societal impacts. The current sustainable development goals for software development corporations, therefore, include addressing the effects of applications use on humans and the environment. The risk maturity model offers a prudent approach to tackling the sustainability challenge in the IT sector, as much as software creation and use are concerned. The risk maturity model allows corporations and governments to trace the progress of threats to sustainability hat each phase of the individual projects (Silvius, Schipper, Van Den Brink, and Planko, 2012). In the context of software development, the risk maturity model can help companies determine the sustainability threats that accompany the development of updates and new software. The end of a software cycle presents the best timing to assess the risks and respond to them accordingly, as the developers seek to create updates or new software (Albertao et al., 2010). Such a move helps address the social and ecological effects of the developments.

The existence of the risk maturity model exhibits the underlying problem with software development, which is the major goal of the practice. Sustainability of business is depicted as a framework for understanding the economic, social, and environmental impact of projects and their management (Martens and Carvalho, 2014). The major challenge of sustainability efforts is the lack of a universal approach to measure and analyze the sustainable extents of a venture. Previously, the creation of computer software arose as a solution to the sustainability woes of IT due to the virtual nature of the applications (Kern et al., 2018). However, it was soon evident that the creation of software was an additional obstacle to IT sustainability due to the wastes in terms of energy and obsolete hardware gadgets. Such devices become obsolete when they fail to support the latest software advancements.

Conclusion

Summarily, sustainable development refers to the creation of businesses and products that are meant to last. Critically, it seeks to address the long-term goals of a company. However, corporations can only realize resilient systems by tackling the effects of their operations on their profits, the community, and the surrounding. Sustainability in software development seeks to incorporate the social, economic, and ecological effects of its activities, as per the triple bottom line approach. Nevertheless, the idea of sustainability in software creation has focused on performing less harm to the environment while maximizing the profits. Most of the activities have ignored the value of the social effects of the advancement, presenting the major sustainability issue of software development. The risk maturity model presents an opportunity to assess the societal effect(s) of previous software before staging updates and new applications. It should serve as the longterm goal of software sustainability because it incorporates the social factor in the effort towards sustainable computer applications.

References

Al Hinai, M. A. and Chitchyan, R., 2014. Social Sustainability Indicators for Software: Initial. *Science,* **79** (68), p. 29.

Albertao, F., Xiao, J., Tian, C., Lu, Y., Zhang, K. Q. and Liu, C., 2010, November. Measuring the Sustainability of Software Projects. In*2010 IEEE 7th International Conference on E-Business Engineering* (pp. 369-373). IEEE.

Beer, M., 2009. High Commitment High Performance: How to Build Resilient Organization for Sustained Advantage. Hoboken: John Wiley & Sons.

Kern, E., Hilty, L. M., Guldner, A., Maksimov, Y. V., Filler, A., Groger, J. and Naumann, S., 2018. Sustainable software products—Towards assessment criteria for resource and energy efficiency. *Future Generation Computer Systems* , **86** , pp. 199-210.

Mahmoud, S. S. and Ahmad, I., 2013. A Green Model for Sustainable Software Engineering. *International Journal of Software Engineering and Its Applications,* **7** (4), pp. 55-74.

Martens, M. L. and de Carvalho, M. M., 2014. *A conceptual framework of sustainability in project management* . Project Management Institute Research and Education Conference.

Oyedeji, S., Seffah, A. and Penzenstadler, B., 2018. Classifying the Measures of Software Sustainability. In Proceedings of the 4th International Workshop on Measurement and Metrics for Green and Sustainable Software Systems co-lacted with 12th International Symposium on Empirical Software Engineering and Measurement (ESEM 2018).

Savitz, A. and Weber, K., 2006. *The Triple Bottom Line* . Hoboken: John Wiley.

Silvius, G., Schipper, R., Van Den Brink, J. and Planko, J., 2012.*Sustainability in Project Management* . London: Gower Publishing, Ltd.

United Nations, 2017. Sustainable Development Goals: 17 Goals to Transform Our World. *United Nations, [Online] Available at <https://www.un.org/sustainabledvelopment/> [Accessed 13 November 2019]* .