Chapter 4
Circular Economy–Driven Software Lifecycle Management: A Sustainable Approach

Usharani Bhimavarapu
https://orcid.org/0000-0001-5050-0415
Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, India

ABSTRACT

Circular Economy (CE) practices aim to achieve maximum utilization of resources, waste reduction, and product life cycle durability, and their application to software lifecycle management became increasingly necessary. The traditional software development has a linear approach, and that's leading to cyclic obsolescence, wasteful utilization of resources, and inefficiency. The paper introduces the application of CE to software lifecycle management for achieving maximum life span of software, modularity, and maintainability. Data were collected using expert questionnaires from general and expert networks, collecting responses on circular economy and bioeconomy principles. Data were preprocessed, and relevant features were extracted using Ant Colony Optimization (ACO) to enhance analytical accuracy. A bi-stacked Long Short-Term Memory (LSTM) network was then used to identify temporal trends in software releases, maintenance records, and usage patterns to offer predictive analysis for anticipatory resource optimization.

DOI: 10.4018/979-8-3373-4652-6.ch004

INTRODUCTION

Circular economy (CE) refers to the approach of reducing waste and optimizing resource utilization in terms of reuse, recycling, and regeneration. The principles of CE in software lifecycle management seek to maximize the utilization of how software is designed, developed, deployed, and retired. Traditional software development is a linear process—develop, use, and throw that creates inefficiency and environmental footprint in terms of energy utilization and hardware dependency. Circular models shatter this linearity focusing on software longevity, modularity, and flexibility. This model conforms to sustainable development goals and business social responsibility for information and communication technology. Underpinned by CE principles, organizations are able to develop software that is robust, upgradable, and compatible. The need is environmental in nature but also through the economic stimulus of cost saving and productivity improvement. Additionally, CE software lifecycle encourages proper utilization of computer resources. It encourages programmers to rethink coding, deployment, and maintenance processes. Lastly, CE-based software lifecycle management minimizes obsolescence and facilitates green computing innovation.

Electronic waste, or e-waste, is a worldwide phenomenon that frequently occurs due to software obsolescence. Obsolete software has the tendency to induce hardware replacement even when physical machinery is in working order. Closed-loop software lifecycle management accomplishes this due to greater backward compatibility and longer software support. By ensuring the software operates on current hardware, organizations reduce hardware obsolescence to be discarded. This lowers the carbon footprint of manufacturing new devices indirectly. Further, modularity in software makes updating or patching easier to update the entire system installations with less storage and bandwidth needs. Software legacy component reuse also minimizes development effort and energy demands. By mapping software onto CE principles, businesses promote environmental sustainability and the lowest cost of operation. Educating developers and IT managers on such practices promotes e-waste reduction. Hence, CE in software prevents not only digital resources but also environmental waste.

Software sustainability is the ability of computer software to stay up and running, efficient, and useful for extended lengths of time. Circular practices focus on developing flexible and sustainable software. This involves constructing modular designs that are simple to extend and upgrade. Sustainable software minimizes the need for full system replacement, saving energy and resources. Further, lifecycle management through CE promotes the use of open-source libraries along with reusable code to minimize repeated effort. Energy efficiency in the domain is also achieved, where better software uses fewer computing resources. Organizations keep low technical