Chapter 3
Circular Economy Approaches to Software Lifecycle Management

P. Selvakumar
https://orcid.org/0000-0002-3650-4548
Department of Science and Humanities, Nehru Institute of Technology, Coimbatore, India

T. C. Manjunath
https://orcid.org/0000-0003-2545-9160
Rajarajeswari College of Engineering, India

Abhijeet Das
https://orcid.org/0000-0003-4599-5462
C.V. Raman Global University, Bhubaneswar, India

ABSTRACT

In recent decades, the world has experienced rapid technological advancement, industrialization, and a significant shift toward digitalization. While these trends have contributed to economic growth and innovation, they have also intensified environmental degradation, resource depletion, and electronic waste. To address these challenges, the concept of a circular economy has emerged as a transformative alternative to the traditional linear economic model of "take, make, dispose." While initially focused on tangible goods and materials, the circular economy model is now expanding to intangible assets, including software. Applying circular economy principles to software development is a novel and promising avenue, aligning the digital realm with sustainability goals. At its core, the circular economy is about

DOI: 10.4018/979-8-3373-4652-6.ch003

designing out waste and maximizing value. One of the foundational principles of circular economy in the software context is design for longevity. This involves writing code that is modular, maintainable, and easy to update or refactor.

INTRODUCTION TO CIRCULAR ECONOMY

In recent decades, the world has experienced rapid technological advancement, industrialization, and a significant shift toward digitalization. While these trends have contributed to economic growth and innovation, they have also intensified environmental degradation, resource depletion, and electronic waste. To address these challenges, the concept of a *circular economy* has emerged as a transformative alternative to the traditional linear economic model of "take, make, dispose." While initially focused on tangible goods and materials, the circular economy model is now expanding to intangible assets, including software. Applying circular economy principles to software development is a novel and promising avenue, aligning the digital realm with sustainability goals. At its core, the circular economy is about designing out waste and maximizing value. One of the foundational principles of circular economy in the software context is *design for longevity*. This involves writing code that is modular, maintainable, and easy to update or refactor. By ensuring that software can evolve with changing requirements and technological environments, developers can reduce the need for frequent rewrites or complete overhauls. Additionally, embracing open-source development, component reuse, and platform interoperability can significantly cut down duplication of effort and promote collective innovation. These practices reflect the "reuse" and "remanufacture" stages of a circular system, whereby software components are not discarded after a single use but are instead reintegrated into new projects or adapted for different contexts. Practices such as lightweight coding, serverless architectures, and cloud resource optimization are integral to aligning software development with circular economy principles. They ensure that digital systems are not only functional and performant but also environmentally responsible.

Beyond technical design, the circular economy also emphasizes *product-as-a-service* models, which can be applied to software through concepts such as Software-as-a-Service (SaaS), platform sharing, and subscription-based licensing. These models shift the focus from ownership to access and value delivery over time, enabling more sustainable consumption patterns. For example, instead of purchasing software that may quickly become obsolete, users can access continuously updated applications through a subscription. This encourages developers to maintain and improve the product over its lifetime, rather than abandoning older versions and pushing for new releases that require complete reinstallation or retraining. Furthermore, the circular