Methodology for the Strategic Re-Engineering of Software Technologies

In the Value Proposition of Sustainable Business Models

Zur Erlangung des akademischen Grades einer
DOKTORIN DER INGENIEURWISSENSCHAFTEN (Dr.-Ing.)

von der KIT-Fakultät für Maschinenbau des
Karlsruher Instituts für Technologie (KIT)
angenommene

DISSERTATION

von

Vera Anja Schott, M.Sc.,
geboren am 09. April 1994 in Kulmbach

Tag der mündlichen Prüfung: 02. Juli 2025
Hauptreferentin: Prof. Dr. Dr.-Ing. Dr. h.c. Jivka Ovtcharova
Koreferentin: Prof. Dr. Rebecca Bulander

Kurzfassung

Softwaresysteme und ihre zugrundeliegenden Technologien erschaffen immer weitere Nutzungspotenziale und werden immer häufiger beim Versuch individuelle Herausforderungen zu lösen herangezogen. Die daraus entstandene Anzahl entwickelter Individualsoftwarelösungen lässt die Softwarelandschaft immer schneller wachsen. Trotz der zunehmenden Erfahrung bleibt die Misserfolgsquote bei der Umsetzung von Individualsoftwarelösungen hoch, was auf erhebliche Herausforderungen bei der effizienten Entwicklung von Softwareanwendungen hinweist. Diese Beobachtung lässt in Zeiten einer stark umweltbewussten und nachhaltig-orientierten Gesellschaft die Frage offen, wie trotz agiler Softwareentwicklungsmodelle und moderner Technologien die Entwicklung von Softwarelösungen nachhaltig gestaltet werden kann, um die bislang beobachtete Misserfolgsquote zu reduzieren. Insbesondere stellt sich die Frage, wie eingesetzte Ressourcen effizient genutzt und ein langfristiger Nutzen für Anwender*innen erzielt werden kann.

Diese Arbeit stellt eine Methodologie vor, die diese Herausforderung durch einen ganzheitlichen Ansatz adressiert. Dabei stehen die Technologie, ihre Aufgabe und der Mensch mit seinen unterschiedlichen Rollen und Verantwortungen im Mittelpunkt. Diese drei, als Ebenen bezeichneten Komponenten, werden unter Berücksichtigung einer nachhaltigen Perspektive analysiert und führen zur Identifikation von sieben zentralen Schlüsselelementen. Da diese Methodologie im industriespezifischen Kontext erstellt wird ist die Erarbeitung von praktischen Leitfäden, die die praktische Umsetzung erleichtern, von hoher Bedeutung. Diese Arbeit schafft durch den dreiteiligen methodologischen Aufbau und den damit verbundenen Schlüsselelementen einen theoretischen Beitrag, während die praktischen Leitfäden einen praxisbezogenen Mehrwert liefern. Schließlich erfolgt die Validierung anhand von herangezogenen Fallbeispielen aus der deutschen Automobilbranche. Das erste Fallbeispiel konzentriert sich dabei auf einen Teilschritt des Recyclingprozesses von Fahrzeugbatterien, während das zweite Fallbeispiel die Problematik von Sondersendungen in der Lieferkette thematisiert. Die durchgeführte Validierung zeigt folglich die Notwendigkeit eines ganzheitlichen Ansatzes über mehrere Ebenen auf und betont die Rolle des Menschen in seinen unterschiedlichen Funktionen vor dem Hintergrund der Entwicklung einer Softwareanwendung für den Einsatz in nachhaltigen Geschäftsmodellen.

Abstract

Software systems and their underlying technologies continue to unlock new potential applications and are increasingly employed to address individual challenges. The resulting proliferation of custom software solutions is driving rapid growth in the software landscape. Despite increasing experience, the failure rate in the implementation of individual software solutions remains high, highlighting challenges in the efficient development of software applications. In an era marked by an environmentally conscious and sustainability-focused society, this raises the question of how software can be made sustainable despite agile development models and modern technologies. Specifically, it calls for addressing how resources can be used efficiently while ensuring long-term value for users.

This thesis presents a methodology that addresses this challenge through a holistic approach. The focus lies on technology, its purpose and the human being with its different roles and responsibilities. These three components, referred to as layers, are analyzed from a sustainability perspective and lead to the identification of seven key elements. As this methodology is developed in an industry-specific context, the development of practical guidelines that facilitate its practical implementation is essential. This work creates a theoretical contribution through the modular methodological structure and the associated key elements, while the practical guidelines add real-world value. Finally, validation is carried out using case studies from the German automotive industry. The first case study focuses on a sub-step of the recycling process for vehicle batteries, while the second addresses special shipments within the supply chain. The validation confirms the incorporation of a holistic approach and emphasizes the role of humans in their various functions in the context of developing a software application for use in sustainable business models.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

24/7	Continuous availability or operation, 24 hours a day, 7 days a week
3D	Three-dimensional
AI	Artificial Intelligence
API	Application Programming Interface
AR	Augmented Reality
BM	Business Model
BMC	Business Model Canvas
CAGR	Compound Annual Growth Rate
cf.	confer
CMM	Capability Maturity Model
CRM	Customer Relationship Management
DEI	Diversity, Equity, Inclusion
DevOps	Development and Operations
DT	Digital Transformation
e.g.	for example
ESG	Environmental, Social, and Governance
GS3M	Generic Sustainable Software Star Model
HCI	Human-Computer Interaction

HR	Human Resource
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
ISO	International Organization for Standardization
ISSPF	Integrated Sustainability Software Performance Framework
IT	Information Technology
MVP	Minimum Viable Product
NATO	North Atlantic Treaty Organization
NFR	Non-Functional Requirement
OEM	Original Equipment Manufacturer
PM	Project Manager
RBCS	Role-Based Control Systems
RQ	Research Question
SDG	Sustainable Development Goals
SPS	Software Project Sustainability
SQA	Software Quality Assurance
SSL	Secure Sockets Layer
SusAD/AF	Sustainability Awareness Diagram / Framework
TPL	Triple Bottom Line
UI	User Interface

UN	United Nations
USD	United States Dollar
USP	Unique Selling Proposition
UX	User Experience
VPC	Value Proposition Canvas
VR	Virtual Reality
XR	Extended Reality

Glossary

Methodology: A structured framework used to systematically analyze, design, and implement processes or strategies to achieve specific objectives.

Re-Engineering: The inclusion of new or additional factors that are crucial during software development and implementation to achieve a specific purpose.

Software Technology: Refers to software applications, software solutions, or digital applications designed to perform specific tasks, solve problems, or provide functionalities in various domains.

Sustainability: The capability to design, develop, and maintain systems in a way that ensures long-term usability, efficiency, and adaptability.

Value Proposition: Added value for users and other stakeholders through the use of the developed software application.

List of Publications

The following publications, developed and published in the context of this dissertation, contribute to its scientific foundation. They are referenced throughout the thesis as Publication I to Publication III. Moreover, it is worth mentioning that Publication III was honored with the **award for the best paper** during the international conference of business and innovative technology (ICBIT 2024).

Publication I	Abdelkafi, O., Ili, S., Schott, V., 2023. Sustainable Success - The Critical Role of Corporate Carbon Footprint in ROI (Study Report). ILI.DIGITAL AG, Karlsruhe, https://ili.digital/boost-roi-through-corporate-carbon-footprint-innovation/
Publication II	Krodel, T., Schott, V., Mayer, A., Ovtcharova, J., 2024. Impact of XR-Enabled Collaboration in Businesses—An Economic, Ecological, and Social Perspective, in: Alareeni, B., Elgedawy, I. (Eds.), AI and Business, and Innovation Research: Understanding the Potential and Risks of AI for Modern Enterprises. Springer Nature Switzerland, Cham, pp. 767–777, doi:10.1007/978-3-031-42085-6_66
Publication III	Schott, V., Ovtcharova, J., 2024. Integrating Sustainability into Software Development: A Global Categorization, in: Mansour, N., Bujosa Vadell, L.M. (Eds.), Finance and Law in the Metaverse World: Regulation and Financial Innovation in the Virtual World. Springer Nature Switzerland, Cham, pp. 59–69, doi:10.1007/978-3-031-67547-8_6

1 Introduction

The age of generative artificial intelligence (AI) triggers a transformation on an intelligent level for the first time, pushes the boundaries of our knowledge, and boosts data-driven decision-making processes. This opens up untapped opportunities for people as the volume of data generated and replicated steadily increases.

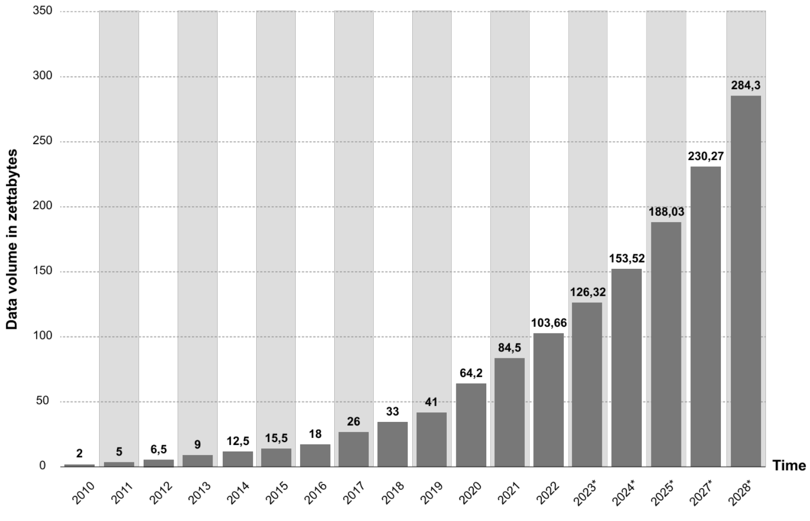


Figure 1.1: Volume of digital data generated/replicated annually worldwide (Tenzer, 2024)

To underscore this observation, Figure 1.1 illustrates the volume of digital data generated annually since 2010 and a projection of its future trajectory through 2028. The data is expected to follow exponential growth over the coming years whereby it is becoming increasingly established as a resource for progress. In contrast, the global demand for natural resources exceeds the earth's regenerating ability. This implies, that while data, a new resource for growth and innovation, can be generated and utilized in seemingly infinite quantities, the availability of physical resources is rapidly diminishing, presenting a notable global challenge. Figure 1.2 exemplifies this observation using the earth overshoot day. It depicts the progression of earth overshoot day from 1971 to 2024, highlighting a consistent trend toward earlier dates each year. This shift reflects an increasing rate of resource consumption, now surpassing earth's annual regenerative capacity by approximately 1.75 times. The global earth overshoot day in 2024 was dated August 1. From this day onwards, humans consume more renewable resources than are available (Club of Rome, 2024).

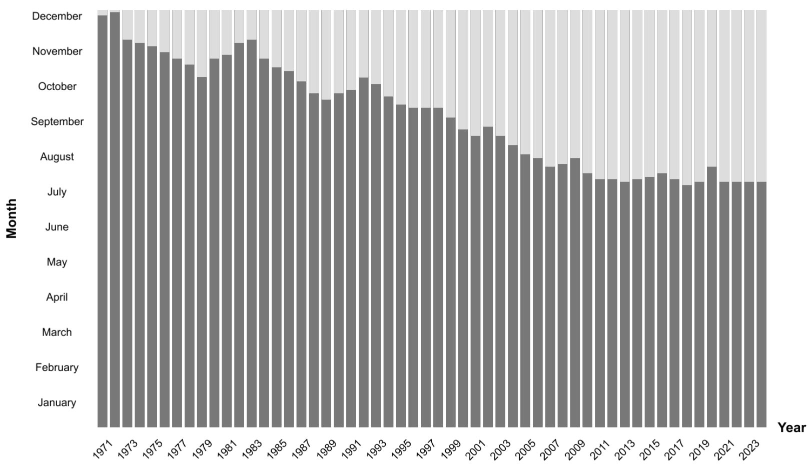


Figure 1.2: Timeline of earth overshoot days (Global Footprint Network, 2024)

The continuously increasing volume of generated and reproduced data is closely interrelated with software technology, which plays a central role in processing, analyzing, and harnessing these data volumes. This relationship is symbiotic: while software technologies are being developed to meet the challenges of the exponential flood of data, the growing volumes of data are simultaneously driving innovation and progress in software development, which in turn generates more data.

In the context of contemporary sustainability, which focuses on conserving resources, enhancing efficiency, and ensuring longevity, a strategic and targeted orientation of new software projects is essential. Such projects must not only meet the increasing demands of data processing but also assume ecological, economic, and social responsibility. Conscious reorientation of software technology can help to promote sustainable solutions and put the symbiosis between data growth and technological development on a future-oriented foundation.

Therefore, the true progress of the 21st century will not only be measured by decision factors such as quality, cost, and time, but will also increasingly be measured by how future-oriented the available technologies will be used to ensure a growth rate across generations in which humanity, nature, and technology are in harmony.

1.1 Motivation

The market is constantly changing, and the cycles of change are becoming increasingly shorter. This means that the economy must adapt more frequently to new circumstances and conditions. Emerging technologies are conquering the market and creating numerous opportunities and possibilities. In addition, transformations are occurring in politics and society, changing the market through new regulations, laws, and behavioral changes. This induces immense pressure on companies, requiring them to keep pace with these developments while continuing to generate profits and margins—a balancing act that determines the dynamics of the market. This shift is likewise recognized by competitive players, requiring a high degree of creativity and reflection on one's own business model (BM) to remain competitive in the long run.

The transition driven by digital transformation (DT) has revolutionized economic interactions. The utilization of mathematical algorithms, AI, and immersive technologies has expanded the range of products and services, transforming them into interactive platforms and ecosystems. As a result, DT has dramatically increased the demand for software projects across all industries in recent years. Current statistics show that the DT market in the manufacturing sector, for example, is expected to grow from USD 307.87 billion in 2023 to USD 733.75 billion by 2028, representing a compound annual growth rate (CAGR) of 18.97% (Silvi et al., 2023). The global forecast across all industries shows a CAGR of 23%-27%, reaching USD 3,289.4 billion by 2030 (Grand View Research, 2024; MarketsandMarkets, 2024). DT and the adoption of software technologies are crucial for future competitiveness, innovation, and the achievement of sustainability goals. However, this transformation requires considerable effort—an effort that is neither self-evident nor simple. Various studies conducted between 2019 and 2022 show a notable failure rate of 66%-78% (Bendor-Samuel, 2019; Garcia, 2022; Robinson, 2019). Ramesh and Delen (2021) reported an error rate of up to 90% in DT projects aimed at increasing efficiency. They also identified five key factors essential for successful transformation: innovation characteristics, thought leaders, diffusion method, timing, and duration.

However, given the high failure rates and increasing awareness of the need to preserve available resources, a critical question emerges regarding the sustainability of current digitalization initiatives.

This leads to another movement influencing the industry: sustainable transformation. It focuses on how globally proclaimed sustainability goals can be achieved. The primary focus lies on the contribution of the economy and the leverage that can be achieved through the active involvement of companies. Therefore, innovations and new BMs are among the core components of both transformation acts. Figure 1.3 illustrates the interconnected roles of DT, sustainability transformation, BMs, and software technology in driving innovation and adaptation. This interconnected system comes with notable complexity

(Penzenstadler et al., 2012). It highlights that each element impacts the others, emphasizing a holistic approach to transformation in business and technology. Nowadays, the connection between digital and sustainable transformation is called twin transformation (Christmann et al., 2024).

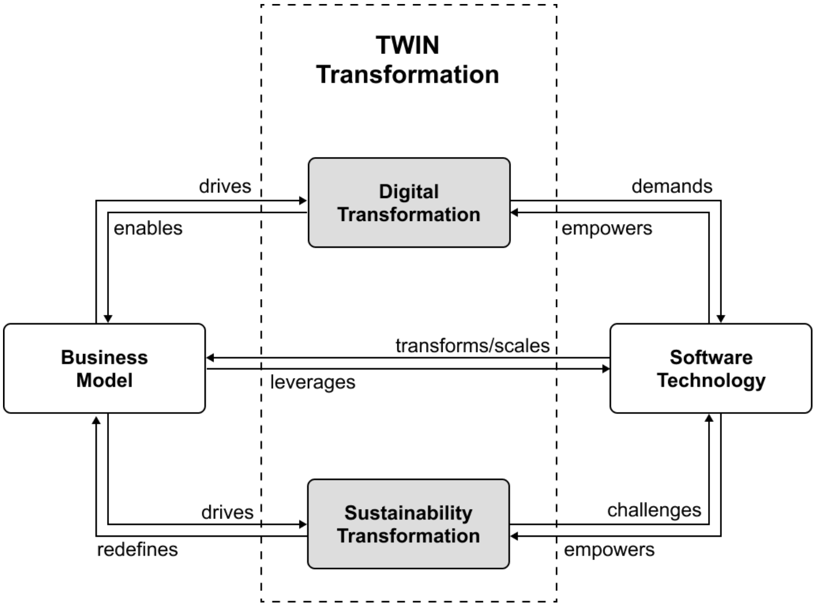


Figure 1.3: Impact diagram of selected transformation drivers

The value of this connection is evident in sectors such as mechanical engineering, where its high relevance stems from the use of both types of resources—data and raw materials. This dual-resource dependency helps explain the sector´s market connection to DT. Data serve as the foundation for DT, while raw materials play a crucial role in processing activities that impact the ecological aspect of sustainability. Furthermore, the manufacturing industry stands as a powerful force in the context of bridging human-machine interactions. Moreover, the engineering sector has long been a center of change and innovation, making it a key driver for transformation across the market, especially for the twin transformation. This work is thus characterized by the integration of several interdisciplinary interfaces across different domains. By focusing on the mechanical engineering industry, this approach bridges modern software technology with a traditional field that faces urgent sustainability challenges. Bringing these innovations into an industry known for its established practices emphasizes the strategic significance of transforming mechanical engineering to meet today's environmental and technological demands. This alignment underscores the potential for profound change at the operational and strategic levels.

1.2 Objectives and Research Question

Building on the tensions and challenges outlined in the previous section, this thesis aims to develop a practical approach for future software projects. The goal is to support these projects in the DT context while addressing the demands of sustainable transformation. This approach should contribute to the efficient and long-term design of software development projects and assist companies in focusing on the possibilities and opportunities that arise when the three topics of software technology, BM, and sustainability are combined. This context is visualized in a Venn diagram (see Figure 1.4), pointing out the identified research field.

Definition of objectives:

This dissertation addresses the identified status quo and the preceding argumentation. It aims to provide an end-to-end methodology for strategically re-engineering software technologies for sustainable BMs. The methodology seeks to contribute to the twin transformation of digitalization and sustainability, helping companies on their way to future growth and adaptation.

Fulfillment of objectives:

The aim of this work is achieved when the methodology developed and validated through use cases is equipped with practical guidelines, making it ready for industry application. Additionally, this work promotes solutions that advance the integration of digitalization and sustainability. The vision for the methodology is not only to combine digitalization and sustainability but also to create a symbiotic, positive impact by addressing both at a holistic level.

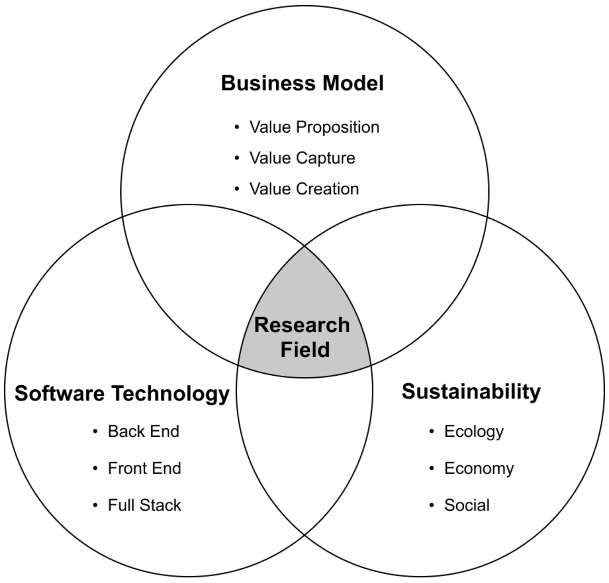


Figure 1.4: Research field

Contribution of objectives:

This research contributes to industry and academia by emphasizing the need to re-engineer software technology strategically to provide a guiding reference for sustainability in DT and strengthen the conceptual basis for future research. In this context, “strategic” refers to the higher-level and decision-making processes to address overarching goals instead of focusing on daily operations and tasks.

The objective of this thesis is to propose a practical methodology to address the challenges mentioned above. The following research questions concretize this methodological endeavor, particularly concerning the re-engineering of software technologies in engineering-intensive industries.

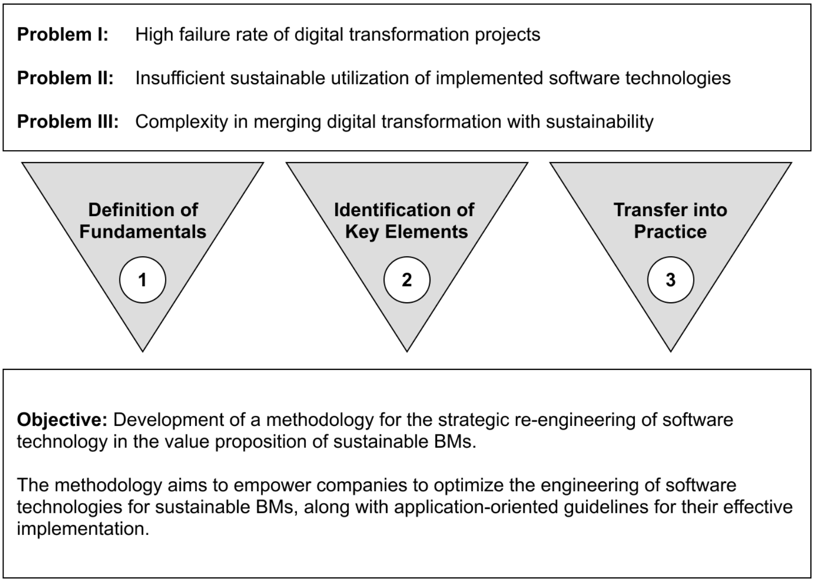

Problem I: High failure rate of digital transformation projects
Problem II: Insufficient sustainable utilization of implemented software technologies
Problem III: Complexity in merging digital transformation with sustainability
Definition of Fundamentals
1
Identification of Key Elements
2
Transfer into Practice
3
Objective: Development of a methodology for the strategic re-engineering of software technology in the value proposition of sustainable BMs.
The methodology aims to empower companies to optimize the engineering of software technologies for sustainable BMs, along with application-oriented guidelines for their effective implementation.

Figure 1.5: Problems, steps and research objective

The first research question pertains to identifying and defining the foundational elements necessary to establish the framework for this approach.

RQ1: How can software technology that balances digital and sustainable transformation be strategically re-engineered?

Based on this, the second research question deals with the key factors that must be addressed to achieve improvement across all three problem statements, as depicted in Figure 1.5.

RQ2: What are the key elements in re-engineering software for sustainable BMs?

Third, translating theoretical insights into practical application steps is often a fundamental element. In this dissertation, a key role is assigned to this process. This is because the orientation toward practical application is central to this work—it is not only about ensuring that the insights are practically implementable but also about how the application level reduces the complexity of the topics being addressed. Given the multifaceted nature of this subject, the focus on practice facilitates more precise understanding and explicitly prepares the gained insights for actual use. Finally, the third research question focuses on turning strategic and theoretical insights into practical application steps.

RQ3: How can the strategic re-engineering approach be adopted into real-world scenarios?

1.3 Thesis Structure

To strategically and effectively master the challenges that are becoming increasingly apparent in the industry, this work endeavors to create a practical methodology. This methodology, which results from the three defined research questions, should serve as a practical guide. To achieve this goal, this thesis follows a seven-step structure. This structure is illustrated graphically comprehensibly in Figure 1.6 and is described below.

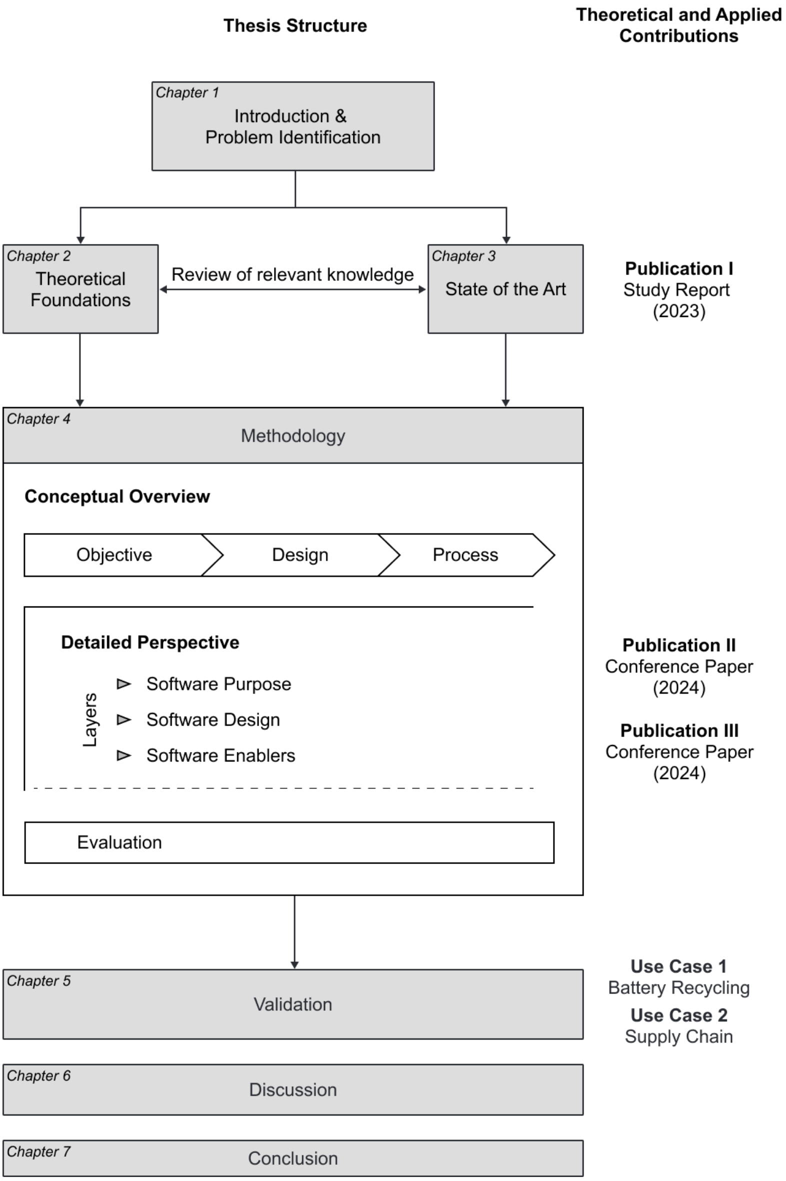


Figure 1.6: Thesis structure

The introductory chapter explains this dissertation's context, outlines the prevailing challenges, and emphasizes the motivation for developing a methodology. To this end, Chapter 1.2 identifies the relevant research areas, clearly defines the study's objectives, and formulates three research questions.

The chapter 'Theoretical Foundations' markedly contributes to a unified understanding and establishes the scientific basis for the entire study. It systematically introduces the relevant research areas—software, sustainability, and BMs—by presenting definitions, models, and developments.

The third chapter, 'State of the Art,' examines the current state of research on integrating sustainability and software technology. Existing models are analyzed and discussed in detail. Approaches of both conceptual and practical nature further refine the relevance of this work. The conclusion of the third chapter outlines the resulting research gap.

Chapter four describes the centerpiece of this thesis. It begins by addressing the objectives of the methodology before outlining its design. In addition to defining the roles involved, the design chapter includes the structure of the basic methodological framework and the detailed architecture consisting of three layers and seven segments. Subchapters 4.3, 4.4, and 4.5 present these layers with their segments and the identified dimensions and influencing factors, and finally translate them into practical guidelines for industrial application. The chapter ends with a proposed evaluation method to classify the extent to which the layers and segments are considered in the re-engineering process.

Chapter five focuses on the methodology validation. It discusses two case studies. Both examples reflect projects in the automotive industry in order to focus on the engineering sector. The first example deals with the recycling process of automotive batteries, while the second case addresses deficits in the supply chain. Both case studies are based on the approach of an individual software solution and address the potential for both digital and sustainable transformation.

After completing the validation, chapter six reviews and summarizes the research questions. In this regard, the focus is on the value contribution achieved for science and industry. Additionally, the discussion chapter concludes with the limitations of the methodology and offers an outlook for further research projects.

The final chapter, “Conclusion”, summarizes the key findings of the thesis in a clear and concise manner.

2 Theoretical Foundations

This chapter concentrates on the key research fields addressed in this thesis, including software technology, sustainability, and BMs.

2.1 Software

The development and use of software drive DT. Therefore, the term “software” describes the programs and applications that run on a computer. In contrast, “hardware” describes the physical components that make up the computer system. Figure 2.1 shows some common examples (e.g., computer monitor as hardware and Microsoft and Android as software). In the past few years, software has fundamentally changed how people communicate with each other and how companies do business. While software technologies were initially viewed as standard supporting tools, they have become the basis for data-driven decisions.

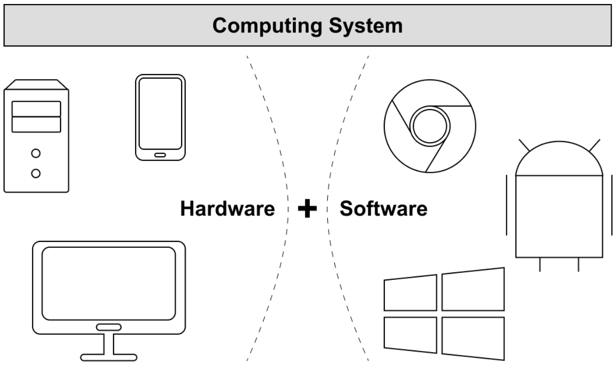


Figure 2.1: Components of the computing system

2.1.1 Fundamentals

Software is generally defined as a program essential for installation, operation, maintenance, and improvement (Humphrey, 1988). It is a collection of instructions designed to execute specific tasks (Hui and Tam, 2002). In a broader sense, Howey describes software technology as any type of concept, process, method, algorithm, or tool that primarily contributes to managing software systems. This understanding not only refers to the technical part but also recognizes the knowledge part embedded within the technology and the expertise needed for effective utilization (Howey, 2002). Moreover, software system design, free from the constraints of physical laws, is often regarded as one of the most unrestricted forms of human creativity (Ousterhout, 2018). As shown in Figure 2.1, hardware and software go hand in hand. Its hierarchical structure is clearly illustrated in Figure 2.2, highlighting the relationship of user, software, and hardware. The illustration from upside down shows that the user interacts with the application software, which in turn uses system software that is runs on hardware. The figure also gives an outline of the layered architecture of a computing system. It emphasizes that the hardware supports system software, enabling application software to perform diverse tasks for end users.

Software plays a critical role in almost all areas of modern life. Accordingly, its importance lies in its ability to automate complex tasks, process data, organize information, enable communication, and extend the functionality of hardware, among other functions. Software is an essential component of DT, driving technological advancements and BM innovations. In recent years, the growing importance of software has led to companies' economic success becoming increasingly dependent on the quality and type of software used and developed.

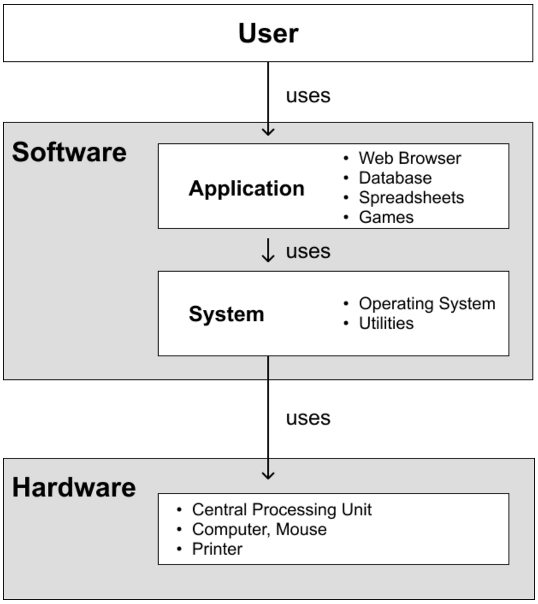


Figure 2.2: Relation between user, software, and hardware, adapted from Wang (2024)

Software Types

As introduced in Figure 2.2, software has different types. The main types are application software and system software. While Figure 2.2 depicts the interdependencies between hardware, system, and application software, Table 2.1 outlines the different categories that differentiate both types of software. The system software underlies the application software, as this ensures that the application software can operate. Application software is therefore responsible for performing specific tasks for the user. It can thus be developed individually, commercially available, or available for free through open-source access or as a cloud native.

This thesis focuses on application software, more precisely on customized application software, also known as bespoke software, that is developed specifically according to the customer's requirements

(Sommerville, 2016). Companies usually choose this type of software for many reasons. Data sovereignty, adaptation to individual work processes, or targeted competitive advantages, for example, by capturing specific customer data, are just a few examples. The highly customizable nature of this type of software also leads to higher risks, longer development times, and additional development costs. Therefore, the methods and characteristics that lead to a successful software application must be achieved with the utmost attention and continuously reviewed and tested.

Table 2.1: Differences between system and application software

System Software	Application Software
Operating System	Individual or Standard
Utility Software	Commercial
Drivers	Open-Source
Middleware	Cloud-Native

Software Quality Attributes

One aspect that leads to successful software applications is the cultivation of software quality attributes. The two pioneering models are one from McCall and the other from Boehm, which built the early foundation of the modern standardized ISO/IEC 25010:2023 quality model. While McCall focuses on allocating 11 selected software quality factors to three software product activities (McCall et al., 1977), Boehm distinguishes between three classifications and seven quality factors (Boehm, 1978). Both concepts contain elements that are still valid but have been enhanced or updated with numerous other models over time, such as Dromey's quality model published in 1995, the FURPS quality model from 1987, and the ISO 9126 software quality model, developed in 1991, which introduces the first international consensus (Al-Qutaish, 2010; Lee, 2014). Table 2.2 compares the five original software quality models and the recommended standardized international version ISO 25010:2023. Each row lists a specific characteristic, and checkmarks indicate which quality models include or emphasize that characteristic.

Table 2.2: Comparison of key characteristics in quality models; based on Al-Qutaish (2010)

Quality models / **Characteristics**	**McCall**	**Boehm**	**Dromery**	**FURPS**	**ISO 9126**	**ISO 25010:2023**
Maintainability	✓		✓		✓	✓
Flexibility	✓					✓
Testability	✓	✓				✓
Correctness	✓					✓
Efficiency	✓	✓	✓		✓	✓
Reliability	✓	✓	✓	✓	✓	✓
Integrity	✓					✓
Usability	✓		✓	✓	✓	✓
Portability	✓	✓	✓		✓	
Reusability	✓		✓			✓
Interoperability	✓					
Human Engineering		✓				
Understandability		✓				✓
Modifiability		✓				✓

Functionality			✓	✓	✓	✓
Performance				✓		✓
Supportability				✓		

According to their consideration in the different models, the most important characteristics for a software are maintainability, efficiency, reliability, usability, portability, and functionality. These quality attributes address different software levels. Attributes such as usability relate to the interface, which is connected to the software´s frontend, while other attributes are more focused on the backend.

Software Components

Frontend and backend are components of a software system, defining two major layers in the software architecture. The frontend layer refers to elements with which the end user directly interacts. A prime example is the user interface, represented by buttons or navigation menus. The frontend is directly related to interactivity, visual aspects, and user experience, summarized as the client side. In contrast, the backend layer concentrates on the underlying functionality of the software and represents the server-side architecture, e.g., application logic and data processing (Amazon Web Services, Inc, 2024). Table 2.3 outlines three further aspects—concurrency, security, and goals—to underline the differences between the frontend and backend.

Table 2.3: Differences between frontend and backend; based on Amazon Web Services, Inc. (2024)

Aspect	**Frontend**	**Backend**
Concurrency	No concurrency needed, because each user has their own copy of the application.	Multiple strategies are available to be prepared for handling a vast number of user requests.

Security	Validation of user input and authentication workflows.	The application of encryption methods and secure coding practices aims to protect databases and services.
Goals	Creating fully functional, responsive and intuitive user interfaces.	Creating reliable architecture that supports frontend development.

Figure 2.3 provides an overview of how the two software layers form a coherent system in which they are different but need to be used in unity. Additionally, the figure provides some examples of programs, such as react, which is used to create the user interface as part of the frontend. Coupling the frontend and backend layers is essential for the functionality of a software system, enabling it to provide a seamless, interactive, and useful experience for the user (Dutonde, 2022).

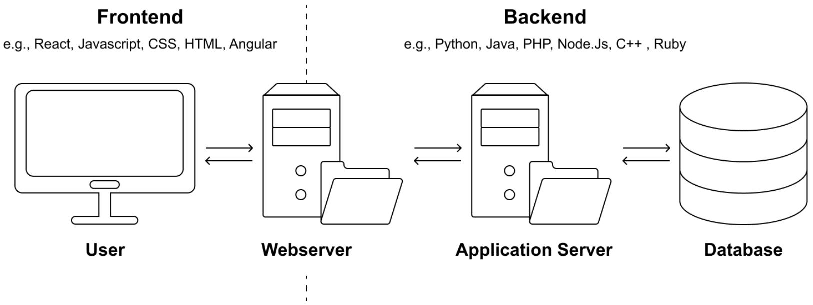

Frontend
e.g., React, Javascript, CSS, HTML, Angular
Backend
e.g., Python, Java, PHP, Node.Js, C++ , Ruby
User
Webserver
Application Server
Database

Figure 2.3: Frontend vs. backend, based on Elser (2022)

2.1.2 Development Models

As indicated in the previous section, software development is a complex, human-driven process that relies on individuals making decisions and judgments, similar to other intellectual processes. In addition, a remarkable degree of creativity is evident, as software systems are considered abstract and are not constrained by physical laws. While this situation can lead to simplification, it can also lead to complexity and difficulty in managing changes (Sommerville, 2016; Wohlin et al., 2015).

However, the ongoing evolution of technology, especially the democratization of generative AI, will cause impactful changes in this field of software development in the coming years. By leveraging AI applications, code generation can be accelerated and made feasible for non-experts, while bug detection can be optimized (Ajiga, 2024; Pothukuchi et al., 2023). This represents only a part of the future possibilities, and researchers worldwide are currently investigating the impact that generative AI will have on areas such as software development.

Table 2.4: Differences between traditional and agile software development methods

Dimension	**Traditional**	**Agile**
Development Process	Linear	Iterative and incremental
Focus	Delivery	Flexibility
Team Size	Large	Small
Requirement	Clear from the beginning	Continuously during the development process
Delivery	Finished product at the end of the process	Interim results during the process
Examples	Waterfall, V-model	Scrum, Kanban, Extreme programming

Sommerville (2016) defined software process as a sequence of four essential activities: software specification, development, validation, and evolution. Software development as a core activity covers multiple stages, starting with the analysis of requirements, followed by design, implementation, test, deployment, and up to maintenance. The natural evolution of models has led to the emergence of various software development models in the literature and practice. Table 2.4 compares traditional and agile software development methods across several dimensions, emphasizing the adaptability and iterative nature of agile versus the structured, linear approach of traditional methods.

The most well-known traditional method is the waterfall method, developed around 1970. This method follows a linear and sequential structure, as depicted in Figure 2.4. Emphasizing the law requiring each phase to be fully completed before the next phase can begin is imperative. This underlines the incremental approach and completely excludes overlaps or iterations, which characterizes the method as very rigid. The phases of the waterfall model match the aforementioned core activities of software development, making this model suitable only for projects with clearly defined and consistent requirements within a stable environment (Sommerville, 2016).

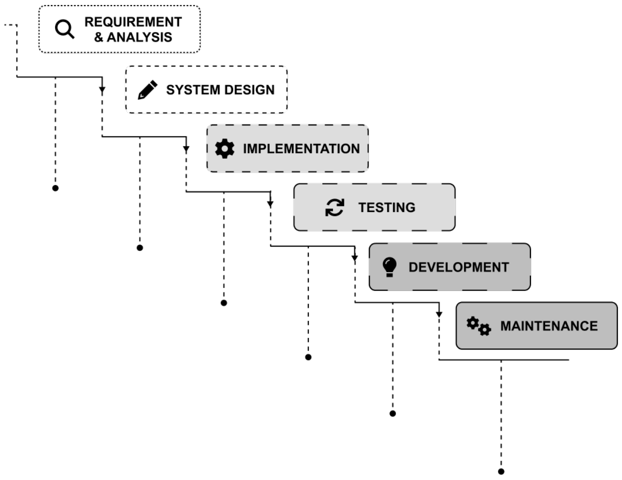


Figure 2.4: Waterfall method

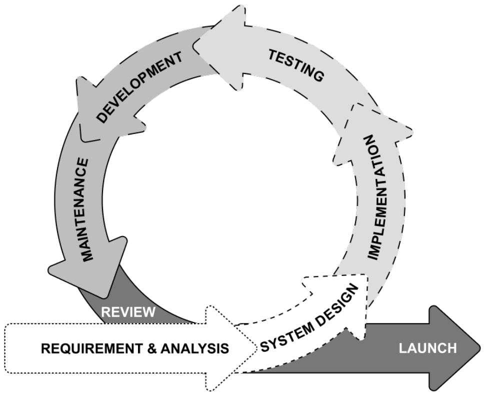


Figure 2.5: Agile method

At the start of the 21st century, with the rise of DT, agile software development become dominant. "Agile" describes a continuous and iterative dynamic, which ensures quick and flexible adjustments and repetition of phases and allows continuous feedback from end users or other stakeholders (Soongpol et al., 2024). This procedure is shown in Figure 2.5. Following this approach, new software versions are created every two to three weeks and delivered to the customer for testing and further feedback. In 2001, a group of software developers presented the manifesto for agile software development, which comprised four fundamental values and twelve principles. The values emphasize (Beck et al., 2001):

- ***Individuals and interactions*** *over processes and tools.*
- ***Working software*** *over comprehensive documentation.*
- ***Customer collaboration*** *over contract negotiation.*
- ***Responding to change*** *over following a plan.*

Sommerville (2016) posited that the core principles of the manifesto can be effectively implemented by fostering active customer involvement, embracing change, emphasizing incremental delivery, maintaining simplicity, and prioritizing people over rigid processes. As mentioned in Table 2.4, agile methods include scrum, kanban, and extreme programming. While methods such as scrum and extreme programming essentially comprise predefined iterations, so-called sprints, the kanban model focuses on continuous processes with defined tasks visualized on boards.

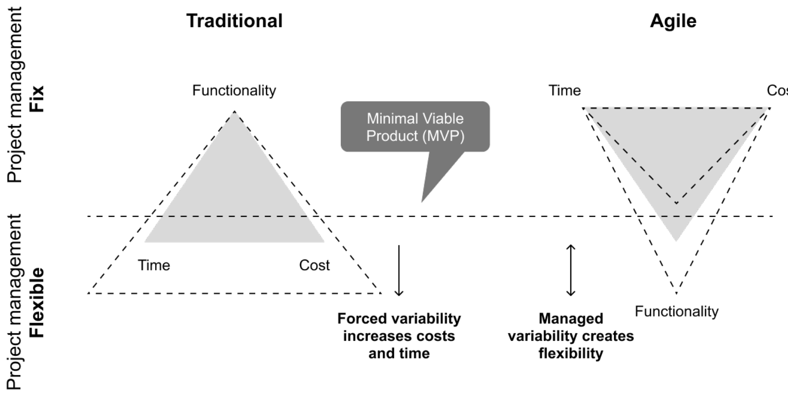

Project management Fix
Project management Flexible
Traditional
Agile
Functionality
Time
Cost
Minimal Viable Product (MVP)
Time
Cost
Functionality
Forced variability increases costs and time
Managed variability creates flexibility

Figure 2.6: The magic triangle according to development methods (Ebert, 2014)

In conclusion, Figure 2.6 highlights the contrast between traditional and agile approaches, indicating that traditional approaches prioritize fixed scope, which leads to higher costs and extended timelines when changes occur. In contrast, agile methods focus on maintaining fixed time and cost, allowing flexibility in functionality to ensure adaptive, timely delivery without compromising efficiency. Agile methods are primarily used in the individual software industry, as they ensure greater transparency and faster adaptability during the development process due to close collaboration, the continuous involvement of the end customer, and the sprint-based development approach. This ensures that the expected functions are delivered accordingly, leading to higher customer satisfaction and quality.

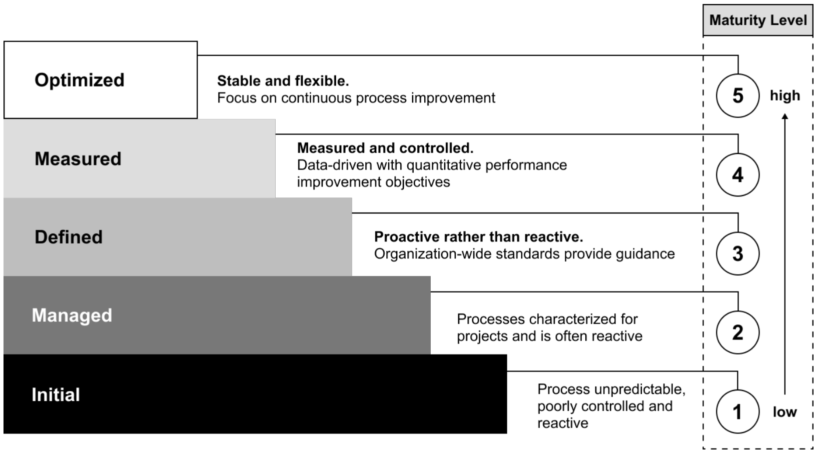

Maturity Level
Optimized
Stable and flexible.
Focus on continuous process improvement
5
high
Measured
Measured and controlled.
Data-driven with quantitative performance improvement objectives
4
Defined
Proactive rather than reactive.
Organization-wide standards provide guidance
3
Managed
Processes characterized for projects and is often reactive
2
Initial
Process unpredictable, poorly controlled and reactive
1
low

Figure 2.7: Capability maturity model, adapted from Welch et al. (2016) and Yassien (2020)

Since both academia and industry are interested in assessing and comparing existing processes and identifying potential areas for improvement, models such as the capability maturity model (CMM) are used. The CMM was initially designed to evaluate the maturity of a software development process (Paulk, 1994) but is now used as an established evaluation model for the performance of software development processes. The model comprises five levels designated in ascending order as follows: initial, managed (*former: repeatable*), defined, measured (*former: managed*), and optimized. Overall, the CMM, as illustrated in Figure 2.7, emphasizes how organizational maturity progresses from reactive to standardized and ultimately to optimized, enabling the enhancement of processes, projects, and broader operational efficiency.

2.1.3 Software Engineering

Having provided key insights into software processes by mainly focusing on software development in Chapter 2.1.2, attention is now shifted to the software engineering scope in Chapter 2.1.3.

Software engineering emerged in response to the so-called "software crisis" of the late 1960s, highlighting the growing challenges of developing reliable, maintainable software systems. As a result, NATO organized the first conference on software engineering in 1968 in Garmisch, Germany, to establish guidelines and best practices for software system development (Sommerville, 2016; Venters et al., 2023). This event marked the formal recognition of software engineering as a distinct discipline within the broader engineering context.

Bourque and Fairley (2014) mentioned that technological aspects not only define software engineering but are also deeply interconnected with aspects such as management and economics, which extend the range of overall considerations.

According to the Institute of Electrical and Electronics Engineers (IEEE), software engineering involves employing structured, methodical, and measurable techniques to design, implement, and sustain software systems. This definition emphasizes the relevance of methods for managing the software lifecycle (Venters et al., 2023).

Sommerville (2016) elaborated on this definition by asserting that software engineering concerns the full range of software production, from system specification to maintenance. He also agreed that this engineering discipline extends beyond technical concerns to include critical areas such as project management and financial considerations.

Therefore, software engineering covers various activities, including the technical, strategic, and organizational aspects of designing, developing, and maintaining software.

2.2 Sustainability

This chapter offers a comprehensive overview of the sociohistorical context of sustainability, encompassing key definitions and a concise explanation of the drivers of its growing relevance in contemporary society. It also examines the relevant legislation and initiatives that promote the development of sustainability practices.

The origin of the term “sustainability” can be traced back to the German forestry industry of the 18th century. Hans Carl von Carlowitz (1645–1714) of Freiberg (Saxony) wrote down the term for the first time in 1713 and has since been considered the father of sustainable thinking. In his work, he advocated for systematic reforestation and the sustainable management of forests to ensure long-term food security and economic viability (Morgenstern, 2007). Since then, the number of definitions and interpretations of sustainability across various contexts has steadily grown. The Oxford English Dictionary (2024) defines the concept of sustainability as a condition “capable of being endured,” “capable of being upheld,” and “capable of being maintained”. Today, the widely accepted core definition of sustainability originates from the Brundtland Report, *Our Common Future*, published in 1987. The definition of sustainability given by the UN World Commission on Environment and Development is also the most often cited worldwide. It defines sustainable development as “development that meets the needs of the present without compromising the ability of future generations to meet their own needs” (United Nations, 1987). Sustainability thus means covering the needs of our time without jeopardizing the needs of future generations. John Elkington developed the triple bottom line framework in 1994, which he later published in *Cannibals with Forks* (Elkington, 1999), to translate theoretical concepts into practical actions. Figure 2.8 shows the trinity approach to ecological, economic, and social sustainability, reflecting the understanding of the 21st century. Notably, the three dimensions are interrelated, intercorrelated, and strongly interdependent. One dimension cannot be fully resolved without addressing the other two dimensions (Dyllick and Hockerts, 2002). Sustainability, therefore, works only if a holistic approach is given and pursued. Each dimension has several key performance indicators.

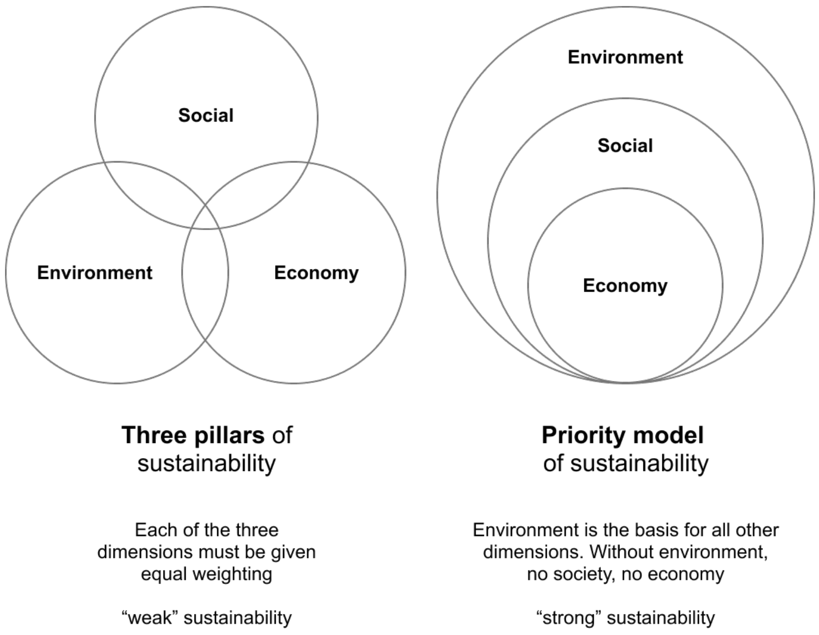

Social
Environment
Economy
Three pillars of sustainability
Each of the three dimensions must be given equal weighting
"weak" sustainability
Environment
Social
Economy
Priority model of sustainability
Environment is the basis for all other dimensions. Without environment, no society, no economy
"strong" sustainability

Figure 2.8: Trinity approach to sustainability, adapted from Miehe et al. (2021)

Thus, the environmental pillar includes waste, energy efficiency and energy consumption, recycling management, raw material use, and water consumption. The social pillar includes education, people and product safety, social well-being, health, and customer satisfaction. The third pillar, economy, covers the aspects of quality, cost, time, and innovation (Assad et al., 2021; Lu et al., 2011). Overall, the concept of sustainability strives to establish a balance between the planet, profit, and people and to maintain this balance in the long term. This goal can be achieved in two different ways, either by assuming that all three elements have the same weighting, as shown on the left-hand side of Figure 2.8, or by promoting the idea that the environment is the basis of all other dimensions and is therefore the mandatory part of creating sustainability, as displayed on the right side of Figure 2.8.

The term “sustainability” shapes political discussions, economic target projects, and social changes. It is considered a paradigm shift, a driver of innovation, and the basis of a new social identity. As shown in Figure 2.9, a whole series of global political milestones on the topic of sustainability began in 1987 with the Brundtland Report. In 1992, at "The Earth Summit", the first World Summit on Sustainable Development in Rio de Janeiro, Brazil, Agenda21 was adopted. The core of this climate framework convention can be found in the underlying principle of *"think globally, act locally"* and includes the goal of stabilizing greenhouse gas concentrations in the Earth's atmosphere. This is intended to prevent, minimize, and mitigate serious negative impacts. However, five years later, the first resolution containing legally binding obligations for reducing greenhouse gases was published. With the Kyoto Protocol, the participating industrialized countries commit themselves to a limitation of emissions for the duration of two commitment periods (1. period: 2008–2012; 2. period: 2013–2020) (Die Bundesregierung, 2022).

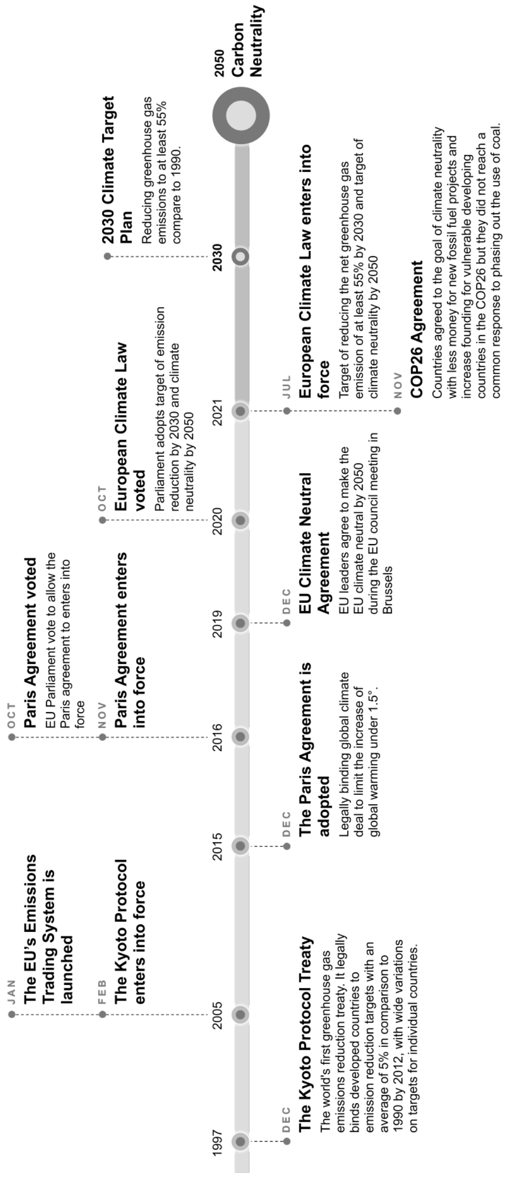

1997
DEC
The Kyoto Protocol Treaty
The world's first greenhouse gas emissions reduction treaty. It legally binds developed countries to emission reduction targets with an average of 5% in comparison to 1990 by 2012, with wide variations on targets for individual countries.
2005
JAN
The EU's Emissions Trading System is launched
FEB
The Kyoto Protocol enters into force
2015
DEC
The Paris Agreement is adopted
Legally binding global climate deal to limit the increase of global warming under 1.5°.
2016
OCT
Paris Agreement voted
EU Parliament vote to allow the Paris agreement to enters into force
NOV
Paris Agreement enters into force
2019
DEC
EU Climate Neutral Agreement
EU leaders agree to make the EU climate neutral by 2050 during the EU council meeting in Brussels
2020
OCT
European Climate Law voted
Parliament adopts target of emission reduction by 2030 and climate neutrality by 2050
2021
JUL
European Climate Law enters into force
Target of reducing the net greenhouse gas emission of at least 55% by 2030 and target of climate neutrality by 2050
NOV
COP26 Agreement
Countries agreed to the goal of climate neutrality with less money for new fossil fuel projects and increase founding for vulnerable developing countries in the COP26 but they did not reach a common response to phasing out the use of coal.
2030
2030 Climate Target Plan
Reducing greenhouse gas emissions to at least 55% compare to 1990.
2050
Carbon Neutrality

Figure 2.9: Timeline of emission regulation (Abdelkafi et al., 2023)

In 2002, the second World Summit on Sustainable Development took place in Johannesburg, South Africa. The meeting, known as "The World Summit on Sustainable Development", not only reviewed Agenda21 but also set some new goals, such as the protection of oceans and biodiversity. This was followed in 2009 by the UN Climate Conference in Copenhagen, Denmark, where no agreement was reached, and no further milestones were set. The subsequent UN Summit Rio+20 ended on 22 June 2012 with the final declaration, "The Future We Want". At this conference, 191 countries worldwide agreed for the first time on a shared understanding of the "green economy" and emphasized the protection of natural resources. In addition, the states decided to define generally applicable sustainability goals in the following years (Bundesumweltministerium, 2020). The initiative culminated in 2015 with the adoption of the Paris Agreement during the UN Summit. Beyond adopting Agenda 2030, the main focus is on the **sustainable development goals (SDGs)** drafted and signed by all UN member states, which aimed at limiting global warming. Compared to the preindustrial level in 1750, global warming should be between 1.5°C and well below 2°C (Die Bundesregierung, 2022). More specifically, the SDGs define 17 goals across all areas of life and serve as a reminder of the need for action. Figure 2.10 presents the official visualization of all 17 goals. Additionally, the 17 SDGs are divided into 169 targets and 232 indicators to concretize the overarching agenda. They have been used worldwide since 2016 to ensure sustainable development and define the roadmap for the future up to 2030 (Bachmann et al., 2022).

Based on this, the European Commission developed **Environmental, Social, and Governance (ESG) factors**, which are intended to support decision makers in acting in a sustainable manner. According to Toniolo et al. (2020), these ESG factors involve environmental issues related to the quality and functioning of natural systems, social issues concerning the rights, well-being, and interests of people and communities, and governance issues pertaining to the management of companies and further associated enterprises. These guidelines are designed to enable companies and entrepreneurs to make sustainable decisions, foster innovation, and align their BMs with sustainability (Toniolo et al., 2020).

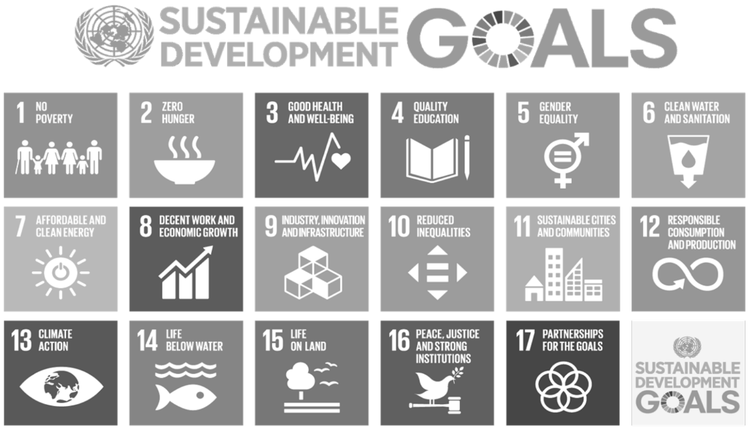

SUSTAINABLE DEVELOPMENT GOALS
1 NO POVERTY
2 ZERO HUNGER
3 GOOD HEALTH AND WELL-BEING
4 QUALITY EDUCATION
5 GENDER EQUALITY
6 CLEAN WATER AND SANITATION
7 AFFORDABLE AND CLEAN ENERGY
8 DECENT WORK AND ECONOMIC GROWTH
9 INDUSTRY, INNOVATION AND INFRASTRUCTURE
10 REDUCED INEQUALITIES
11 SUSTAINABLE CITIES AND COMMUNITIES
12 RESPONSIBLE CONSUMPTION AND PRODUCTION
13 CLIMATE ACTION
14 LIFE BELOW WATER
15 LIFE ON LAND
16 PEACE, JUSTICE AND STRONG INSTITUTIONS
17 PARTNERSHIPS FOR THE GOALS
SUSTAINABLE DEVELOPMENT GOALS

Figure 2.10: SDG17 (United Nations, 2015)

Market research suggests that future markets for sustainable products and services will likely be huge. In fact, the United Nations SDGs are expected to create market opportunities of over USD 12 trillion per year by 2030 (Elkington, 2018). Europe intends to play a pioneering role by introducing the European Green Deal, including a climate neutrality commitment by 2050 (BMZ, 2024). Sustainability is thus an omnipresent challenge with great economic opportunities. A megatrend of the 21st century.

2.3 Business Model

In today's rapidly evolving digital landscape, technology alone is no guarantee of innovation and long-term success. The real competitive advantage lies in transforming technological innovation into tangible economic value embedded in a BM.

2.3.1 Design

The term “business model” has gained enormous popularity in the past decades due to the age of digitalization and worldwide accessibility to the internet in science and business. The term has become a fad, often misinterpreted as a buzzword or interchanged with neighboring terms. Without delving into the detailed literature discussion, a BM and a corporate strategy must clearly be distinguished, as they are often treated as equivalents. This work, therefore, agrees with Osterwalder´s (2004) opinion that strategy, BMs, and process models all address similar challenges but solve them on different levels and through various approaches. Figure 2.11 illustrates the three business layers, depicting their hierarchical order while simultaneously indicating the specific output achieved at each level. This classification should, therefore, specify that the BM is assigned to an architectural role through which it is clearly defined how turnover is generated.

The following section presents a selected chronological overview of the most frequently cited definitional approaches, offering a structured insight into the key definitions used in the literature.

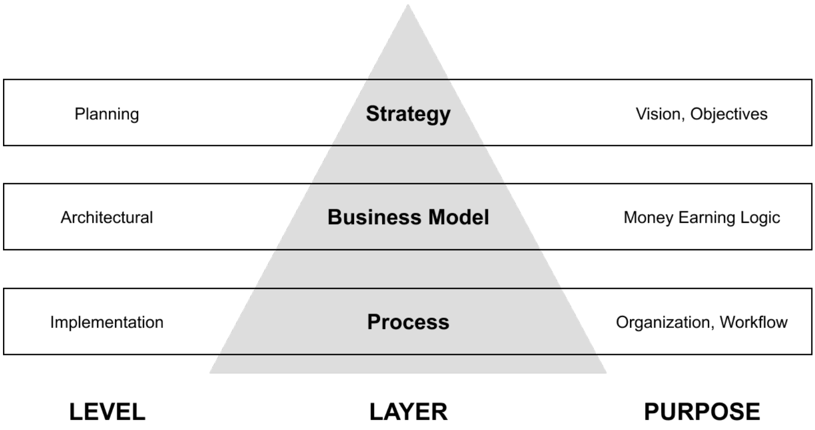


Figure 2.11: The three business layers (Osterwalder, 2004)

Timmers (1998) formulated a three-part definition stating that a BM is "an architecture for the product, service and information flows, including a description of the various business actors and their roles; and a description of the potential benefits for the various business actors; and a description of the sources of revenues". In this context, Timmers states that a BM is based on an exchange between at least two different parties through which potential advantages can be achieved and revenues are generated.

Amit and Zott (2001) summarized that a BM is a precise description of the construct of how to add value by exploiting business opportunities. The authors further emphasized that the BM outlines how transactions are structured, detailing their content, organization, and governance mechanisms.

Chesbrough et al. (2002) first introduced their concept of a BM in 2002, defining it as a practical framework that links technical potential to economic value by engaging with customers and markets.

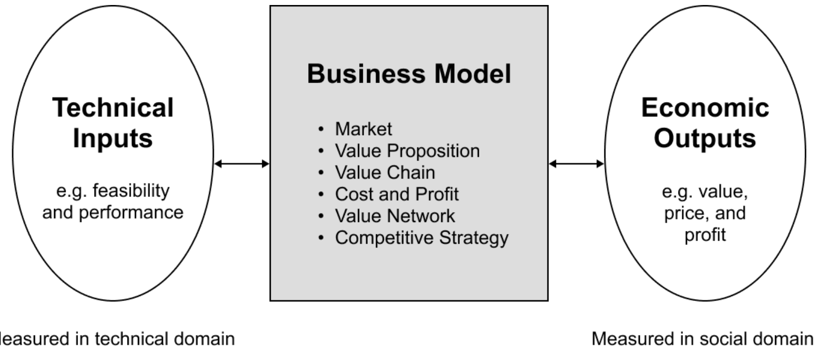


Figure 2.12: The BM mediates between technical and economic domains (Chesbrough et al., 2002)

As shown in Figure 2.12, this definition considers the understanding of a BM to be the mediator between technological feasibility and economic return and value. Thus, the BM acts as a bridge translating technical potential into market opportunity by defining how a company creates, delivers and captures value. Understanding the BM is essential for any company that strives to leverage technological advancements and maintain its leading edge in the marketplace (Chesbrough et al., 2002). Therefore, the key difference in the definition is the integration of a technological component.

For practical implementation, Chesbrough et al. (2002) further recommended answering the following six core elements while creating a BM:

(1) articulate the value proposition

(2) identify a market segment

(3) define the structure of the value chain

(4) estimate the cost structure and profit potential

(5) describe the position of the firm within the value network

(6) formulate the competitive strategy

A classic approach for designing a BM involves using the so-called magic BM triangle depicted in Figure 2.13. It describes the four key elements of a BM comprising three essential cornerstones and a centerpiece. The BM triangle is thus framed by the questions of what, how, and why. The question "What?" refers to the value proposition, the question "How?" to the value creation, and the question "Why?" to the fundamentals of value capture. At the core, all three elementary corners unite to form the question "Who?", which in turn refers to who is to be addressed by the BM. It defines the target and customer audience (Gassmann et al., 2017).

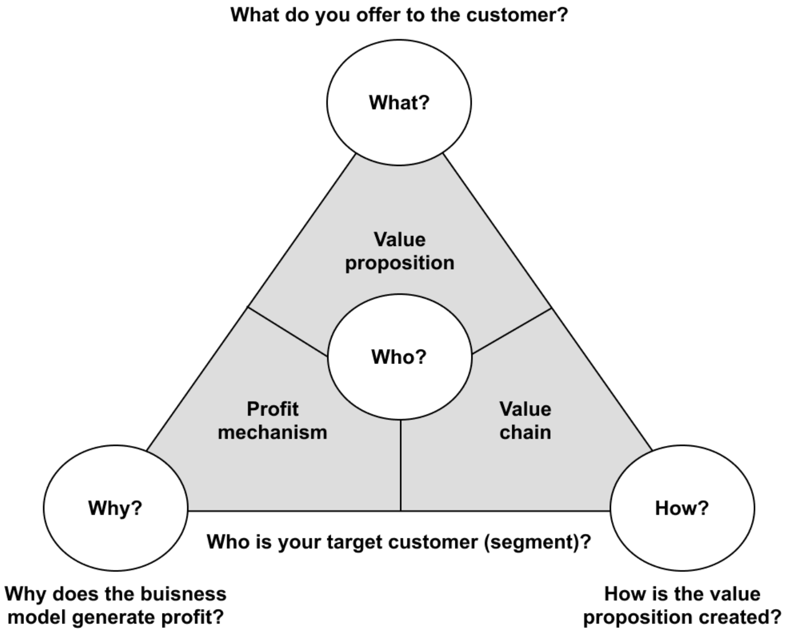
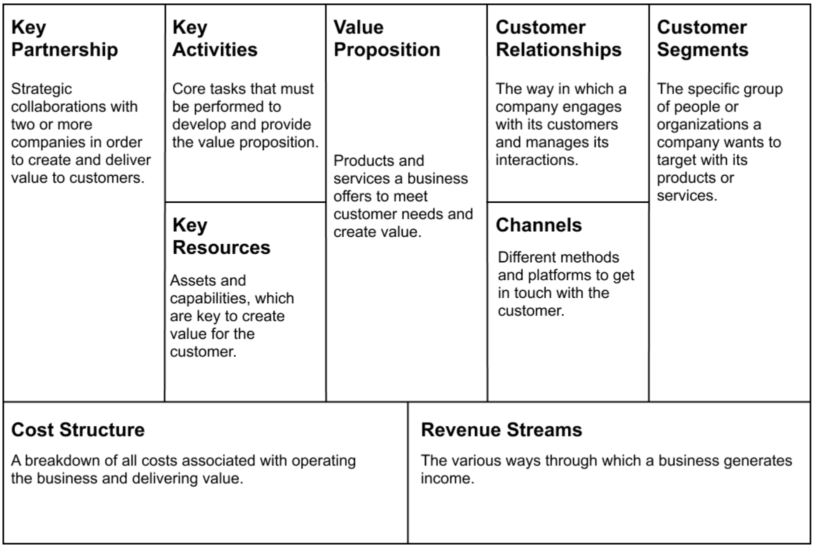

What do you offer to the customer?
What?
Value proposition
Who?
Profit mechanism
Value chain
Why?
How?
Who is your target customer (segment)?
Why does the buisness model generate profit?
How is the value proposition created?

Figure 2.13: The BM triangle (Gassmann et al., 2017)

Nevertheless, the best-known and practice-wide definition of a BM originates from Osterwalder's dissertation in 2004. Its widespread popularity is based on the practical representation of a BM expressed as a canvas. Using this approach, Osterwalder points out that a BM is a conceptual tool that comprises several elements and their relationship to each other. Through this, a company visualizes how it generates profits. In detail, a BM "(...) is a description of the value a company offers to one or several segments of customers and the architecture of the firm and its network of partners for creating, marketing and delivering this value and relationship capital, in order to generate profitable and sustainable revenue streams" (Osterwalder, 2004). In essence, a BM explains the logic behind how an organization creates, delivers, and captures value (Osterwalder and Pigneur, 2010). This categorization becomes very clear in the above-mentioned BM canvas (shown in Fig. 2.14) and has served as a reference and point of reference since its creation, both in practice and in theory.

The canvas comprises nine core elements. The center addresses the value proposition. The left side focuses on internal aspects required to create value, such as key activities, key resources, and key partners. In addition, the right side concentrates on the external dimensions necessary to describe the way value is delivered to the customer. Related fields are customer relationships, channels, and customer segments. To complete the canvas, the two elements at the bottom—cost structure and revenue streams—contribute to the BM aspect of value capturing.

Reflecting on the development of the definition of a BM, the focus has increasingly shifted from value capture solely to the broader dimension of value creation (Zott et al., 2011).



Key Partnership
Strategic collaborations with two or more companies in order to create and deliver value to customers.

Key Activities
Core tasks that must be performed to develop and provide the value proposition.

Key Resources
Assets and capabilities, which are key to create value for the customer.

Value Proposition
Products and services a business offers to meet customer needs and create value.

Customer Relationships
The way in which a company engages with its customers and manages its interactions.

Channels
Different methods and platforms to get in touch with the customer.

Customer Segments
The specific group of people or organizations a company wants to target with its products or services.

Cost Structure
A breakdown of all costs associated with operating the business and delivering value.

Revenue Streams
The various ways through which a business generates income.

Figure 2.14: The BM canvas, adapted from Osterwalder (2004)

In a nutshell, the proposed work will be based on the following definition: a BM describes the way in which a company creates added value for its customers and generates revenues. It requires an outline of the value proposition, value creation and value capturing. The value proposition describes the benefits and added values that customers or other parties can gain from this company. Value creation architecture describes the way in which benefits are generated. Value capturing defines how a business generates revenue and, ultimately, how it makes money (Ili, 2009).

2.3.2 Value Proposition Dimension

According to Osterwalder, the value proposition describes the specific benefits that targeted customer groups gain upon adopting a company's products or services. It outlines the unique value a company offers, which sets it apart from its competitors. This differentiation is the fundamental reason why customers choose one company over another, as it directly addresses their needs or problems in a way that competitors do not. Thus, the value proposition serves as a key driver of customer decisions and loyalty (Osterwalder, 2004).

Establishing a strong value proposition necessitates a) identifying the pain points of the chosen customer segment, b) observing the target market as well as its dynamics and influencing factors, and c) analyzing the competition and identifying the unique selling proposition (USP). This USP is the driving differentiator and key to achieving business success through clear and transparent communication. Ensuring sustained success requires reviewing and refining the value proposition on a regular basis and adapting it to a vibrant market (Brunner et al., 2024).

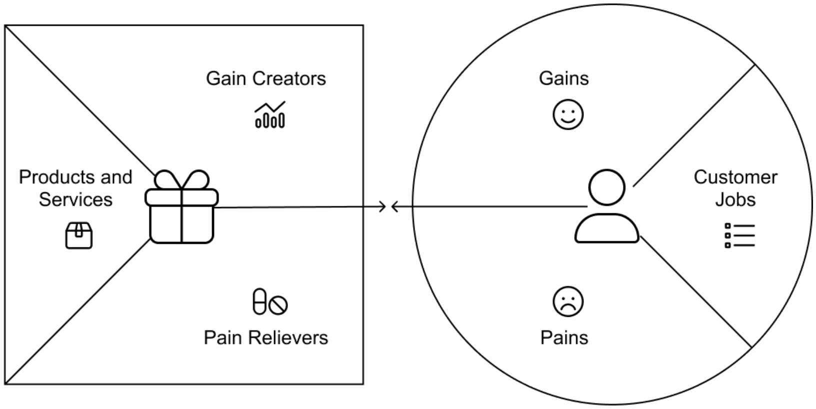


Figure 2.15: Value proposition canvas (Osterwalder et al., 2015)

The value proposition may operate at the intersection of the company's offering, its communication strategy, and the customer's perception. It reflects the alignment between the company’s message and the customer’s understanding of the product or service's value and is crucial for shaping the overall effectiveness of the value proposition (Belz, 2006).

For effective implementation, the following questions offer guidance in defining a value proposition that aligns with customer needs (Grothus et al., 2021):

- What needs and/or problems does the customer have?
- What challenges does the customer face? And how can the company help the customer to master these challenges?
- For which service is the customer willing to pay?

Interviews, surveys, creative sessions (e.g., brainstorming), and observations are helpful in answering these and similar questions. The insights gained can then be used to compile the value proposition canvas according to Osterwalder and Pigneur. The canvas, shown in Figure 2.15, comprises the left-hand and right-hand sides. While the left side focuses on the customer, the right side spotlights the product or service. To be more precise, this means that information about needs, tasks, wishes, and challenges is needed to formulate the customer side and that insights about features and functions, for example, are key to providing a complete understanding of the product (Osterwalder et al., 2015).

2.3.3 Digital and Sustainable Impact

Analog BMs are characterized by value creation that operates without any digital influence, focusing solely on the sale of a physical product. Traditional stationary retail is a perfect example (Chesbrough, 2010; Teece, 2010). Technological advances and changing societal expectations markedly influence the transition from traditional to digital and sustainable BMs. DT enables companies to automate processes, collect data, and thus adapt their digital offerings. Societal demands for sustainability are also forcing companies to rethink the way they do business, considering their sustainable footprint.

Digital BMs have created an entire generation of new BMs through DT. Customers are offered essential added value, which is based on digital technologies and generates new revenue streams. In practice, the two types of digital BMs are **platform-based** and **data-based**. A platform BM also known as a digital marketplace, is defined by the interaction of supply and demand on a platform to create value. The more users meet, the higher the value of the platform. Different platforms include product, service, and development types. In contrast, data-based BMs are called big data businesses. The core task is the processing of collected data into information, knowledge and finally digital intelligence, as visualized in Figure 2.16. It is used by companies to optimize their performance or to adapt their entire value proposition to the needs of their customers. In practice, however, this detailed analysis of operational and user data is usually not only for optimization purposes but also for selling and trading between companies (Grothus et al., 2021).

Data
Information
Knowledge
Intelligence

Figure 2.16: The power of data

Steininger (2018) followed another approach of classifying the different levels of digital BMs and introduced the following types:

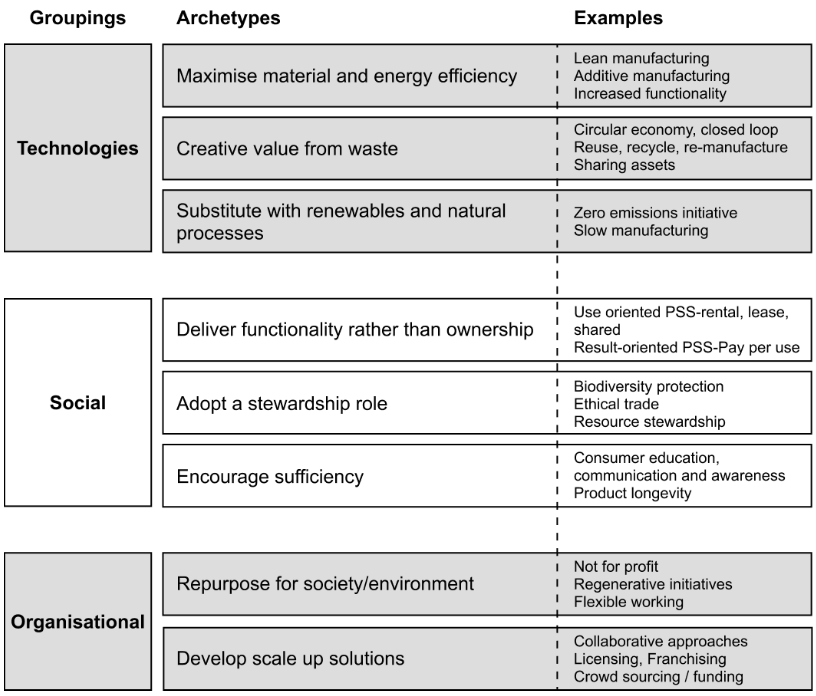
1. **IT-facilitated BMs**: Technology is used in the infrastructure.
2. **IT-mediated BMs**: Technology is used in the infrastructure and customer interfaces.
3. **IT-bearing BMs**: Technology is used in the infrastructure and with product pillars.

4. **Digital BMs**: Technology is used in all aspects, from infrastructure to customer interface to product offers.

For example, Amazon can be defined as a platform- and data-based BM, according to Grothus et al. (2021), while it can also be defined as an IT-mediated to digital BM, depending on whether a digital product or a physical product is ordered by the customer, when following the approach of Steininger (2018).

Two more world-famous digital BMs are Uber and Airbnb. In both cases, the disruptive character of these models is highlighted, as they have one thing in common: they revolutionized their industries entirely without owning any physical end products. Uber became the most successful taxi ride provider without owning a single vehicle. Airbnb has become one of the largest holiday rental providers and does not own a single property. Digital technologies form the basis because added value is a globally accessible digital platform that becomes the arena for offer and demand. Users can either offer a service themselves, i.e., a taxi ride or a place to stay, or meet a desired offer there in real time and from anywhere. Digital technologies allow humans to satisfy their needs anytime and from any place and respond to countless offers. According to Weill and Woerner (2013), smooth interactions between content, experience, and platform is essential in digital BMs to provide customers with a compelling offer. Therefore, the success of digital BMs relies on transforming user data into information, which is then used to derive knowledge and build intelligence (see Fig. 2.16).

Moreover, sustainable BMs are defined beyond the familiar definition of BMs because they contribute to the company's and society's positive development. Lüdeke-Freund (2010) defined this additional level as the "superior customer value". Therefore, sustainable BMs still aim to identify solutions that allow companies to capture economic value. Concurrently, these offers also focus on fulfilling non-financial targets, meaning generating environmental and social value to follow the triple bottom line principle (Schaltegger et al., 2012). Examples of sustainable BMs are presented in Figure 2.17, where Bocken et al. identified a total of eight archetypes categorized into technological, social, and organizational groups.



Groupings	Archetypes	Examples
Technologies	Maximise material and energy efficiency	Lean manufacturing Additive manufacturing Increased functionality
	Creative value from waste	Circular economy, closed loop Reuse, recycle, re-manufacture Sharing assets
	Substitute with renewables and natural processes	Zero emissions initiative Slow manufacturing
Social	Deliver functionality rather than ownership	Use oriented PSS-rental, lease, shared Result-oriented PSS-Pay per use
	Adopt a stewardship role	Biodiversity protection Ethical trade Resource stewardship
	Encourage sufficiency	Consumer education, communication and awareness Product longevity
Organisational	Repurpose for society/environment	Not for profit Regenerative initiatives Flexible working
	Develop scale up solutions	Collaborative approaches Licensing, Franchising Crowd sourcing / funding

Figure 2.17: Sustainable BM archetypes, adapted from Bocken et al. (2014)

2.4 Conclusion of Chapter 2

In this chapter, the theoretical foundations covering the disciplines of software technology, sustainability, and BMs were examined. Thereby, the focus was on selected aspects relevant to this thesis's scope. The theoretical foundations of software were based on presenting the key elements. In addition to a broad understanding structure, the subchapter referred to different development models and clarified the meaning and scope of software engineering. Focusing on sustainability, the main emphasis was on development and its increasing relevance. In particular, a brief outline of the political milestones, such as the SDG17 agenda, was provided. Finally, Section 2.3 introduced the topic of business models. The design components were presented, highlighting the value proposition of BMs. The basics of BMs were described after the issues of software and sustainability to demonstrate the effects and associated changes of the digital and sustainable transformation of BMs in Chapter 2.3.3. The entire second chapter was based on the results of a comprehensive literature review, which primarily comprised scientific works and relevant websites.

3 State of the Art

Regarding the previous principles and the objectives of this thesis, Chapter 3 presents the state of the art and is organized in three sections. In this context, Sections 3.1 and 3.2 describe and discuss the latest insights regarding the topic of re-engineering and sustainability in software engineering. Section 3.3 discusses various approaches published between 2011 and 2024.

3.1 Relevance of Re-Engineering

The relevance of re-engineering is just as timely and required nowadays as it was at its birth in the early 1990's. While the original intention of re-engineering was to reduce costs, especially in the phases of increased service and maintenance work (Sneed, 1991), today's aim goes beyond the cost factor. It increasingly incorporates the three pillars of sustainability, meaning that alongside the cost factor, factors related to people and the environment are becoming equally relevant.

Software re-engineering generally refers to the process in which existing software or parts are reworked and reused in a new system. The process of re-engineering starts with analyzing the existing software, followed by designing to improve it, and finally, integrating it into the new system. The newly developed system often replaces the old one entirely while saving time and resources (Baumöl et al., 1996).

This underlying idea of re-engineering is familiar not only in technical processes but also in business processes. Hammer and Champy (1993) defined re-engineering in the area of business processes and stated that re-engineering means "the fundamental rethinking and radical redesign of business processes to achieve dramatic improvements in critical, contemporary measures of performance, such as cost, quality, service, and speed". Indeed, business process re-engineering (BPR) can be seen as a strategic, interdisciplinary activity that needs to be paired with the characteristics of management for success (Dey, 1999).

Software re-engineering is primarily motivated by the need to update fundamental software technology or to enhance business operations (Sneed and Verhoef, 2001). Moreover, various reasons for considering software re-engineering include the following five most common ones (Sneed, 1991):

- Reducing maintenance costs
- Decreasing the error rate
- Converting the software to a better platform
- Extending the life of a system
- Enabling business change within the company

After identifying the main drivers for the decision to re-engineer, considering the potential challenges that may arise during the process is equally important. The five critical challenges teams and organizations will likely face when implementing rethinking and redesigning initiatives are listed below (Dey, 1999).

1. Adapting to a paradigm shift requires time.
2. Implementing new tools and techniques necessitates capital investment.
3. Advocating for the new process needs support within the team.
4. Gaining relevant experience requires training and practice.
5. Ensuring effective communication requires improved coordination.

After outlining the motivations and challenges of the re-engineering process, emphasizing the need for successful project re-engineering through optimal integration of technical and human factors becomes crucial. Furthermore, Dey (1999) explained that the human factor is intensely versatile, as it includes elements such as a clear corporate vision, new values, an adapted culture, leadership, and innovative skills Figure 3.1 presents four criteria and related sub-criteria that are decisive for Dey to achieve seamless integration. These include culture, structure, process, and technology. In sum, this alignment is essential to ensure that technological advances are effectively enabled by human engagement and organizational transformation.

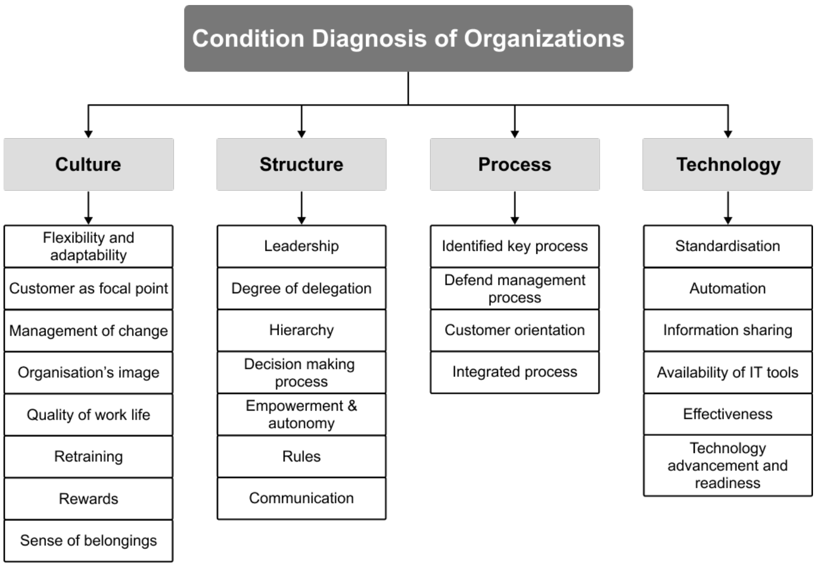


Figure 3.1: Key criteria for seamless integration of re-engineering systems, adapted from Dey (1999)

As mentioned at the beginning of this chapter, nowadays, the relevance of re-engineering is equally important. However, it must be emphasized that the reasons given in the context of the goal to reduce maintenance costs are no longer sufficient to address the current complexities, nor are they target-oriented enough to meet the evolving demands of the organization. Based on this perspective and for the purposes of this dissertation, this thesis defines the objective of re-engineering as adding a sustainability layer to software design, development, and implementation.

3.2 Sustainability in Software Engineering

Numerous definitions and neologisms that have emerged in recent years, mainly since 2010, aiming to link the topics of sustainability and software markedly increase the complexity of this field. Therefore, recognizing the cornerstones of this research stream (for fundamentals, see Chapter 3.2) is crucial. Building on this foundation, clearly defining the target project and distinguishing it from existing work (for differentiation, see Chapter 3.3) becomes essential.

This thesis argues that definitions of terms such as “sustainable software”, “software sustainability” and “sustainability in software (engineering)” share a common aim of integrating the underlying idea of sustainability presented in Chapter 2.2 into software technology. The existing literature has yet to clarify which dimensions of sustainability apply in the field of software. While some research adopts the approach of the classic triple bottom line principle, focusing on the economic, ecological, and social dimensions, other authors opt for an extended model that incorporates additional dimensions such as technical and individual (Condori-Fernandez et al., 2019; Penzenstadler and Femmer, 2013). This extended model focuses specifically on the design aspects of sustainability. By expanding the scope to include technical and individual dimensions, a more holistic approach to software sustainability is achieved. However, this expansion introduces additional layers of complexity, making it more challenging to maintain a clear focus in research and practice. In addition, the extended dimensions may have emanated from the original three dimensions, as they often intersect with ecological, economic, and social concerns. Furthermore, sustainability dimensions are not isolated. Instead, they interact and influence one another, creating a dynamic system in which changes in one dimension affect the others.

The concept of software sustainability is multi-layered, as various definitions prove. One example is Koziolek's definition, which was formulated in the initial phase around 2011 and in which sustainable software is characterized as “a software-intensive system that operates for more than 15 years” or as “a long-living software system which can be cost-efficiently maintained and evolved over its entire life-cycle” (Koziolek, 2011). He also argues that sustainability implies the core characteristics of maintainability, modifiability, transferability, and evolvability. While Koziolek's approach is straightforward, it predominantly reflects the technical perspective and does not consider broader factors. The emphasis on the technical aspect also appears in other definitions. Thus, Penzenstadler defines software sustainability as “preserving the function of a system over a defined time span” (Penzenstadler, 2013). Likewise, the definition from the Software Sustainability Institute (2024) states that sustainability in software is achieved when the “software you use today will be available - and continue to be improved and supported in the future”. The more definitions emerge over time, the more complex it becomes to determine what is and is not declared sustainable. Similarly, numerous definitions are evident in the literature dedicated to specific sub-areas or aspects. For example, there is an increasing view that sustainable software is a non-functional requirement and expands the existing list of quality attributes set out in ISO/IEC 25010 as presented in Chapter 2.1.1 (Calero et al., 2013; Khalifeh et al., 2023; Venters et al., 2014).

Furthermore, software sustainability can be referred to as the ability to design, develop, deploy, and maintain reliable, long-lasting software that meets the specific needs of users (Acar, 2017). This emphasizes the importance of a customer-focused approach, ensuring that the software is tailored to user requirements throughout its lifecycle. Therefore, alongside the dimensions of time and function, the quality aspect (including the aspect of maintenance) and the focus on the customer can be identified as further key factors for defining software sustainability.

A further step in clarifying the topic involves distinguishing between sustainability in the software process and sustainability in the software product. Both elements must undoubtedly be examined in the context of sustainability in order to effectively commit to sustainable practices in software engineering (Khalifeh et al., 2023; McGuire et al., 2023). In detail, software product sustainability refers to outcome sustainability. This can be the software itself or the added value the software can provide upon usage. This part includes factors such as software features and functionalities, its maintainability, and its impact on the environment, economy, and society over its lifecycle (Khalifeh et al., 2023). Conversely, software process sustainability pertains to the sustainability aspects involved in creating the software. This includes, for example, management activities and the development of the software itself.

Besides the subdivision into software product and process sustainability, a very popular approach derived from the IT sector is equally used as a reference. It presents the two perspectives of “Green in IT” and “Green by IT.” While the IT-focused model only considers the environmental aspect of sustainability, the software engineering model is intended to consider the triple bottom line approach. Penzenstadler (2013) thus introduced the so-called absolute definition, which concentrates on software for sustainability, and a relative definition, which focuses on the sustainability aspects within the software engineering process. Oyedji et al. (2018) referred to this approach and presented a structured overview clearly defining the respective points of contact for sustainability in the field of software engineering. Overall, they identified four task areas presented in a Venn diagram (see Figure 3.2). The four perceptions comprise three independent areas and one area defined by the intersection of all three other areas. The main perceptions are thus assigned to the development, usage, and focus impact, while the intersection area reflects the net effect. Following this segmentation, sustainability in software development integrates sustainable practices during design and development. Software for sustainability targets the purpose software can add to achieve sustainability goals. Green software systems aim to reduce environmental impacts and promote green awareness. The intersection presents the total impact and suggests a holistic view in which sustainability efforts are applied in individual software systems and across a larger ecosystem, incorporating elements from all three surrounding areas (Oyedeji et al., 2018).

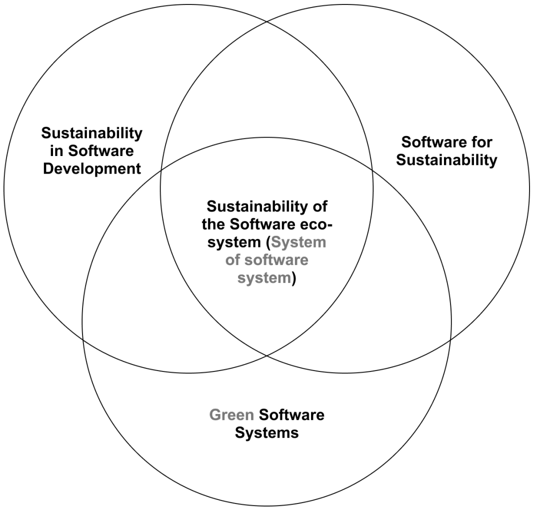

Sustainability in Software Development
Software for Sustainability
Sustainability of the Software eco-system (System of software system)
Green Software Systems

Figure 3.2: Categorization of software sustainability perceptions (Oyedeji et al., 2018)

Valid for all four perceptions, the motivation to place sustainability in software engineering arises from different angles, as outlined by Lammert et al. (2024b). First, they stated that reducing the negative impact on the environment is a common driving factor without further discussing whether this motivation is extrinsic or intrinsic. They also mentioned the desire for long-lasting software as a reason for considering sustainable factors in software engineering. Furthermore, sustainability is often seen as a risk factor in many companies, as it is linked to important aspects such as reducing costs, protecting image and reputation and positioning the company as an attractive and modern employer. These factors play a decisive role in the economic success and competitiveness of a company and are therefore motivators for achieving sustainability (Lammert et al., 2024b).

Sustainability and software engineering are already independent of each other, two highly complex topics, but both can drive innovation and create new BMs, especially when merged. The exploration of various definitions and the recognition of differing perspectives on how and where sustainability can be implemented demonstrate the need to consider both technical and non-technical factors for a systematic and holistic approach to the challenge (Imran and Kosar, 2019). To make it tangible, Imran and Koser listed technical and non-technical aspects in their work, which can primarily be assigned to the timeframe during and after software development. Figure 3.3 shows the proposed list, with a total of three technical (design, code, and user experience) and one non-technical focus.

In conclusion, the process of sustainable software engineering can be described as a multilayered and continuous effort. The associated interdisciplinary requirements pose organizational challenges that require the use of various tools and techniques, as well as the integration of change management methods within the corporate culture (Atadoga et al., 2024).

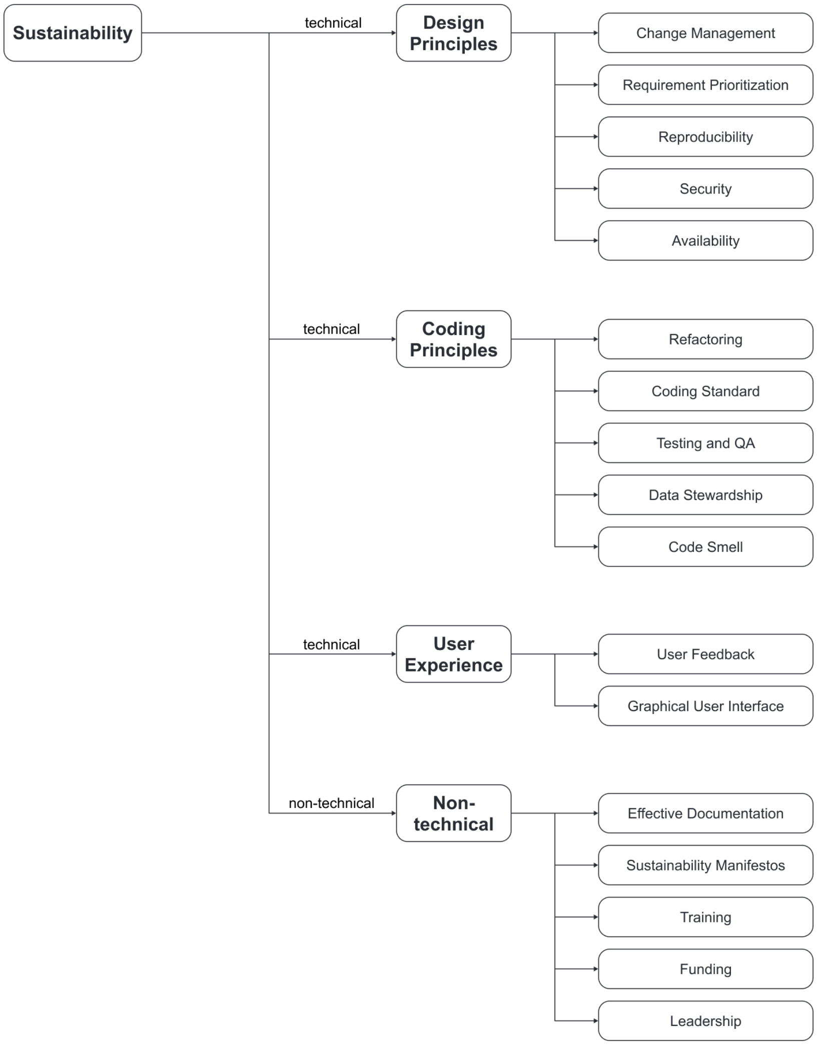


Figure 3.3: Different aspects of software sustainability, adapted from Imran and Kosar (2019)

In this thesis, the three core dimensions of sustainability–economy, ecology and social–are used to integrate sustainability in software engineering, combined with the conviction that technical and non-technical aspects are relevant. In addition and as described before, the following two characteristics are key for understanding software sustainability:

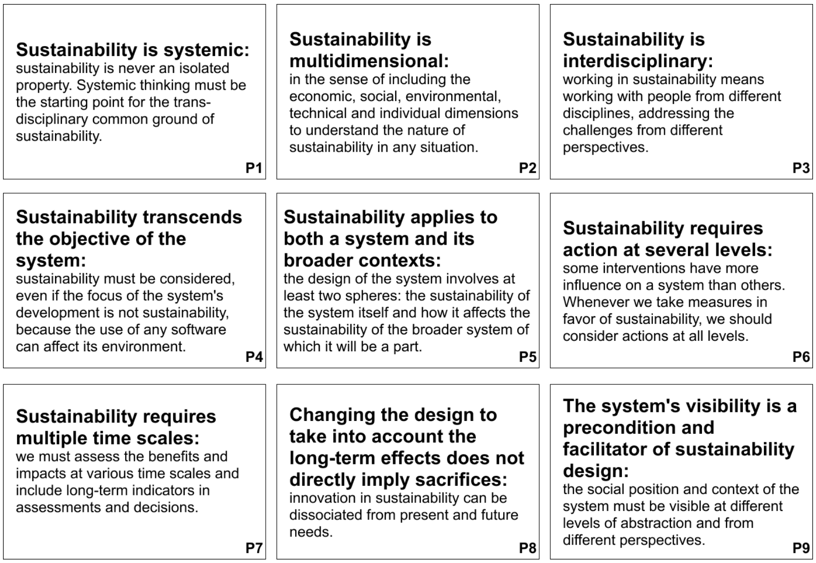
1. **Longevity through customer-centricity:** The software should be used continuously by the user due to its strong acceptance, ongoing relevance, and lasting value.

2. **Longevity through system quality:** The software should ensure stability and functionality through continuous improvements, maintenance, and enhancements of the system and its functions.

3.3 Related Frameworks from the Literature

The relevance of this topic, combined with the high complexity resulting from diverse perspectives, has underscored the need for practice-oriented models since the beginning of this research stream around 2010. These approaches vary widely in nature. Some models are limited to addressing a single sustainability dimension (ecological, economic, or social) or focus exclusively on either a technical or non-technical analysis. Others, by contrast, follow the software process, concentrating within their frameworks and methods specifically on the design and development phases of the process. From 2010 to 2024, numerous highly detailed research initiatives have been conducted, with an increasing trend toward greater complexity and comprehensiveness in the models developed in recent years. Beyond studies published by individual researchers or small working groups, new international research teams have continuously emerged. The most well-known cooperative achievement in sustainable software research is the Karlskrona Manifesto from

2014, developed during the 22nd IEEE International Requirements Engineering Conference in Karlskrona, Sweden. Christoph Becker laid the groundwork for this project by posing the question, “Are sustainability and longevity two sides of the same quality?” (Becker, 2014). Ten experts, including well-respected and frequently cited scientists such as Birgit Penzenstadler and Colin C. Venters, collaborated on this work.



Sustainability is systemic: sustainability is never an isolated property. Systemic thinking must be the starting point for the trans-disciplinary common ground of sustainability. **P1**

Sustainability is multidimensional: in the sense of including the economic, social, environmental, technical and individual dimensions to understand the nature of sustainability in any situation. **P2**

Sustainability is interdisciplinary: working in sustainability means working with people from different disciplines, addressing the challenges from different perspectives. **P3**

Sustainability transcends the objective of the system: sustainability must be considered, even if the focus of the system's development is not sustainability, because the use of any software can affect its environment. **P4**

Sustainability applies to both a system and its broader contexts: the design of the system involves at least two spheres: the sustainability of the system itself and how it affects the sustainability of the broader system of which it will be a part. **P5**

Sustainability requires action at several levels: some interventions have more influence on a system than others. Whenever we take measures in favor of sustainability, we should consider actions at all levels. **P6**

Sustainability requires multiple time scales: we must assess the benefits and impacts at various time scales and include long-term indicators in assessments and decisions. **P7**

Changing the design to take into account the long-term effects does not directly imply sacrifices: innovation in sustainability can be dissociated from present and future needs. **P8**

The system's visibility is a precondition and facilitator of sustainability design: the social position and context of the system must be visible at different levels of abstraction and from different perspectives. **P9**

Figure 3.4: Principles of the Karlskrona manifesto, adapted from Silveira et al. (2022)

The Karlskrona manifesto serves to realign the software engineering research field toward the question of sustainability and establishes nine overarching principles, which are listed in Figure 3.4.

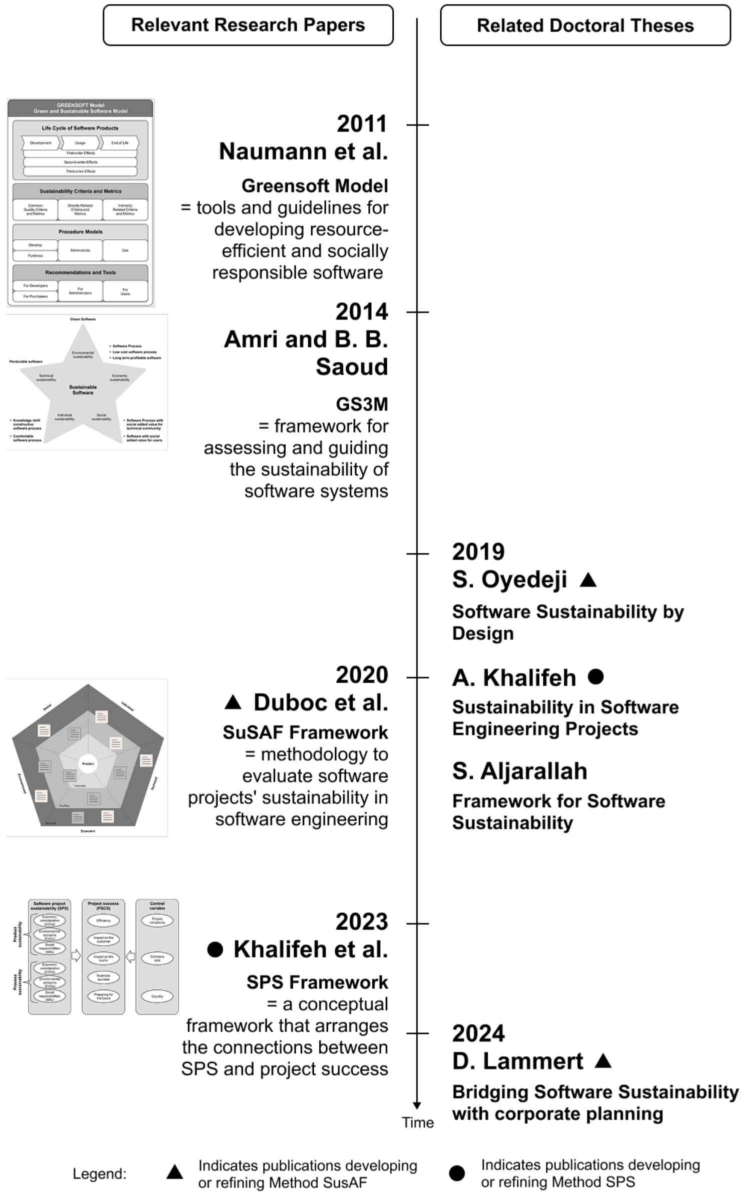

Relevant Research Papers
Related Doctoral Theses
2011
Naumann et al.
Greensoft Model
= tools and guidelines for developing resource-efficient and socially responsible software
2014
Amri and B. B. Saoud
GS3M
= framework for assessing and guiding the sustainability of software systems
2019
S. Oyedeji ▲
Software Sustainability by Design
2020
▲ Duboc et al.
SuSAF Framework
= methodology to evaluate software projects' sustainability in software engineering
A. Khalifeh ●
Sustainability in Software Engineering Projects
S. Aljarallah
Framework for Software Sustainability
2023
● Khalifeh et al.
SPS Framework
= a conceptual framework that arranges the connections between SPS and project success
2024
D. Lammert ▲
Bridging Software Sustainability with corporate planning
Time
Legend:
▲ Indicates publications developing or refining Method SusAF
● Indicates publications developing or refining Method SPS

Figure 3.5: Overview of previous related scientific research projects

After a thorough review of the available frameworks, five models from eight studies are presented. These eight studies comprise four scientific papers and four doctoral dissertations. Figure 3.5 provides an overview of the central thematic focuses, arranging the studies according to their publication dates. As the legend in Figure 3.5 indicates, the symbol triangle and circle mark connections and interrelationships between the studies to enhance clarity and traceability.

The following informed individual assessments guided the selection of these five frameworks. The first framework, the Greensoft model, was chosen because it is widely accepted as a foundational approach for integrating sustainability into software development. The second model, referred to as the Generic Sustainable Software Star Model (GS3M), is considered one of the earliest approaches to incorporating expanded sustainability dimensions, including both technical and individual aspects. The software project sustainability (SPS) model was included because it interestingly connects sustainability dimensions with overall project success, providing a perspective that extends beyond the direct aspects of sustainability.

Several years later, the development of application-oriented frameworks began. Two models that focus on practical applications are the Software Sustainability Framework by Aljarahlla and the widely recognized Sustainability Awareness Framework (SusAF). The following sections introduce the five selected models.

Greensoft Model (Naumann et al., 2011)

The Greensoft model by is a conceptual reference model that considers the entire software lifecycle and various stakeholders and activities, describing their respective influences. The model plays a crucial role in integrating sustainability across different phases, providing a framework applicable to a broad range of users, from developers to end users.

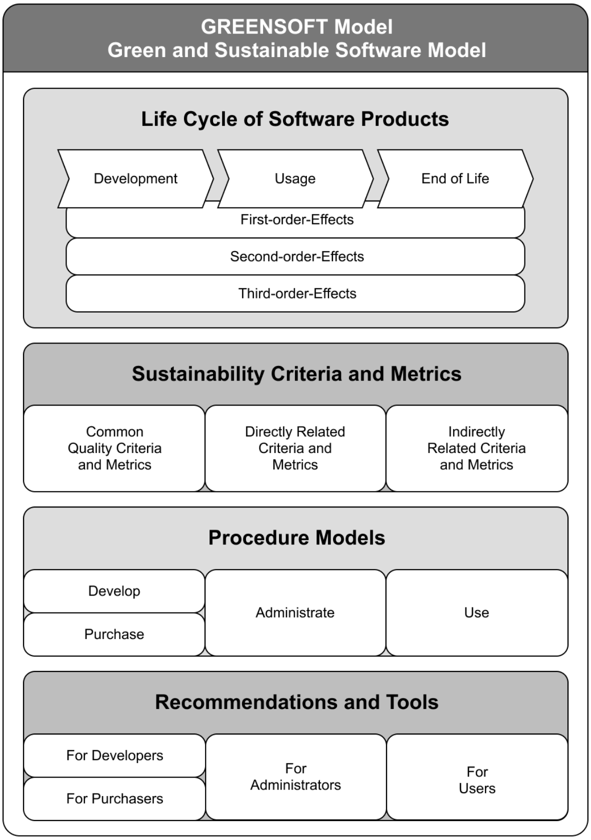

GREENSOFT Model
Green and Sustainable Software Model
Life Cycle of Software Products
Development
Usage
End of Life
First-order-Effects
Second-order-Effects
Third-order-Effects
Sustainability Criteria and Metrics
Common Quality Criteria and Metrics
Directly Related Criteria and Metrics
Indirectly Related Criteria and Metrics
Procedure Models
Develop
Purchase
Administrate
Use
Recommendations and Tools
For Developers
For Purchasers
For Administrators
For Users

Figure 3.6: Greensoft model (Naumann et al., 2011)

As illustrated in Figure 3.6, the Greensoft model comprises four parts. It initially addresses the sustainability assessment of the software product throughout its lifecycle in the "Life Cycle of Software Products," exploring the impacts of software on the environment and society. In this regard, part two, sustainability criteria and metrics, provides an approach for measuring software quality. Direct and indirect metrics are considered, which are further divided into first-to third-order criteria. Part three, the procedure model, proposes the provision of appropriate guidelines and tools to promote a greener approach to software design and implementation. Part four, recommendation and tools, looks at guidelines and best practices for sustainable implementation. Accordingly, while part three focuses on methodological approaches to developing and managing sustainable software, part four provides concrete tools for practical implementation (Naumann et al., 2011).

Overall, the Greensoft model is highly process-oriented and operates at a significant level of abstraction, which presents challenges for its direct practical implementation (Condori-Fernandez and Lago, 2019).

Generic Sustainable Software Star Model (Amri and Bellamine Ben Saoud, 2014)

The GS3M is a conceptual approach that combines the five-dimensional understanding of sustainability according to Penzenstadler and Femmer (2013) while merging the concept of software product and software process in one model.

The model aims to identify the characteristics and attributes at each level that contribute to defining and achieving sustainable software. As shown in Figure 3.7, the five sustainability dimensions are represented in a star-shaped arrangement. Attributes such as resource efficiency and energy consumption are associated with ecological sustainability, while intellectual capital aligns with the economic dimension, and factors such as job satisfaction pertain to the individual dimension. Technical quality attributes, such as maintainability and usability, support the durability of the software, while the social dimension is enhanced by attributes such as accessibility and communication. In essence, this model considers software sustainable if the software product and development process meet specific key attributes. The software product is considered sustainable, as it is ecofriendly, durable, socially beneficial for users, and offers long-term economic value. Similarly, a sustainable development process should contribute positively to the technical community, support the growth of organizational intellectual capital, be cost-effective, foster developers' knowledge and skill enhancement, and offer a comfortable working environment. Collectively, these characteristics ensure that the product and process align with broader sustainability goals (Amri and Bellamine Ben Saoud, 2014).

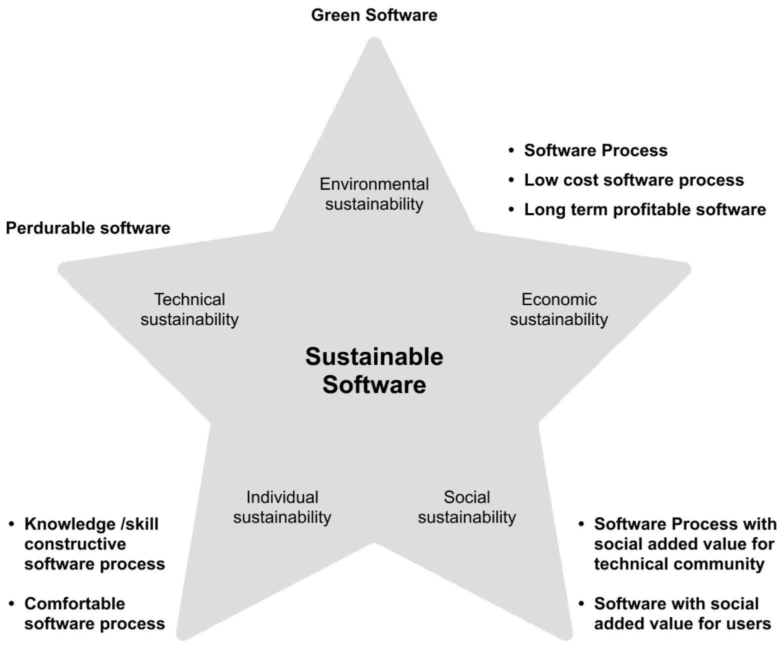

Green Software
Environmental sustainability
• Software Process
• Low cost software process
• Long term profitable software
Perdurable software
Technical sustainability
Economic sustainability
Sustainable Software
Individual sustainability
Social sustainability
• Knowledge /skill constructive software process
• Comfortable software process
• Software Process with social added value for technical community
• Software with social added value for users

Figure 3.7: Generic sustainable software star model (Amri and Bellamine Ben Saoud, 2014)

Reflecting on this model, it aims toward building a general level of awareness of what needs to claim software as sustainable, while it provides no practical guideline or support on how to implement.

Software Project Sustainability and Project Success (Khalifeh et al., 2020, 2023)

Khalifeh et al. (2023) noted, within the scope of a research project, that the relationship between sustainability and project success in the context of software has yet to be extensively studied. This combination is described as SPS. The proposed model aims to contribute to this field by exploring the potential connections and influencing factors between these two areas.

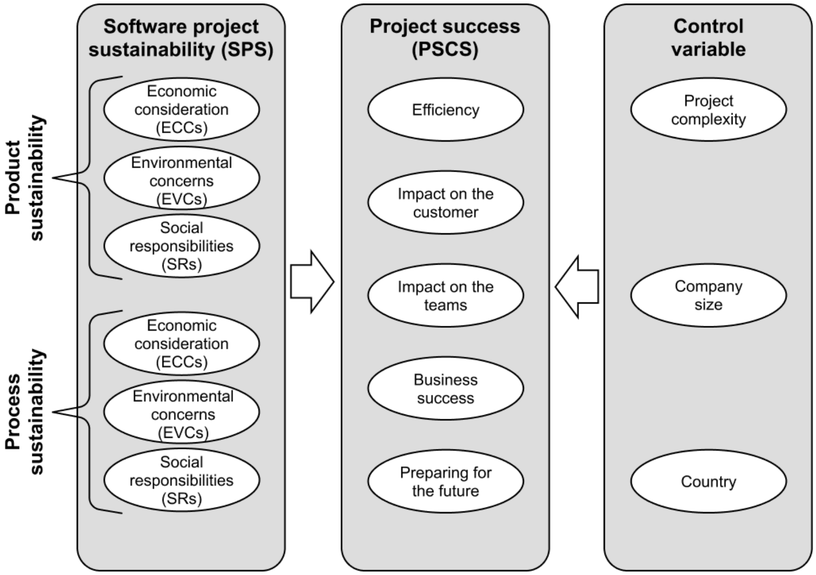

Software project sustainability (SPS)
Product sustainability
Economic consideration (ECCs)
Environmental concerns (EVCs)
Social responsibilities (SRs)
Process sustainability
Economic consideration (ECCs)
Environmental concerns (EVCs)
Social responsibilities (SRs)
Project success (PSCS)
Efficiency
Impact on the customer
Impact on the teams
Business success
Preparing for the future
Control variable
Project complexity
Company size
Country

Figure 3.8: Software project sustainability and project success (Khalifeh et al., 2023)

As shown in Figure 3.8, the proposed framework comprises three components: SPS, project success (PSCS), and control variables. The first component, SPS, addresses the commitment to and consideration of all three sustainability dimensions within the software product and software process. The sustainability dimensions related to the software product refer to non-functional aspects, such as maintainability and security. Along the process, variables such as energy consumption are considered. The foundation for this framework was established in the 2020 study, “Incorporating Sustainability into Software Projects: A Conceptual Framework” (Khalifeh et al., 2020).

The second and middle part of the concept relates to project success and nominates the five elements of efficiency (time and cost management), customer impact, business success, team impact, and preparing for the future (e.g. innovation and adaptability). The last and third parts of the framework add control variables that can influence the relationship between sustainability and project success. Among them are factors such as company size, country, and project complexity. The underlying theoretical foundations of this framework were first comprehensively presented in Khalifeh's doctoral dissertation (Khalifeh, 2020), while the elaboration and visualization have been published in a scientific article (Khalifeh et al., 2023).

Indeed, the framework represents a valuable theoretical contribution to the relevant research area and adds further aspects to the discussion, as it integrates project success criteria. However, a more detailed examination of the framework is still required, as evidence of step-by-step implementation in real use cases remains scarce.

A Framework for Software Sustainability (Aljarallah, 2021)

A further framework, presented in Figure 3.9, which is intended to combine the topics of sustainability and software, was proposed in the doctoral thesis by Sulaiman Aljarallah. He chooses a holistic approach and integrates the ecological, economic, social, political, and technical dimensions. These are labeled as D1 through D5 in Figure 3.9.

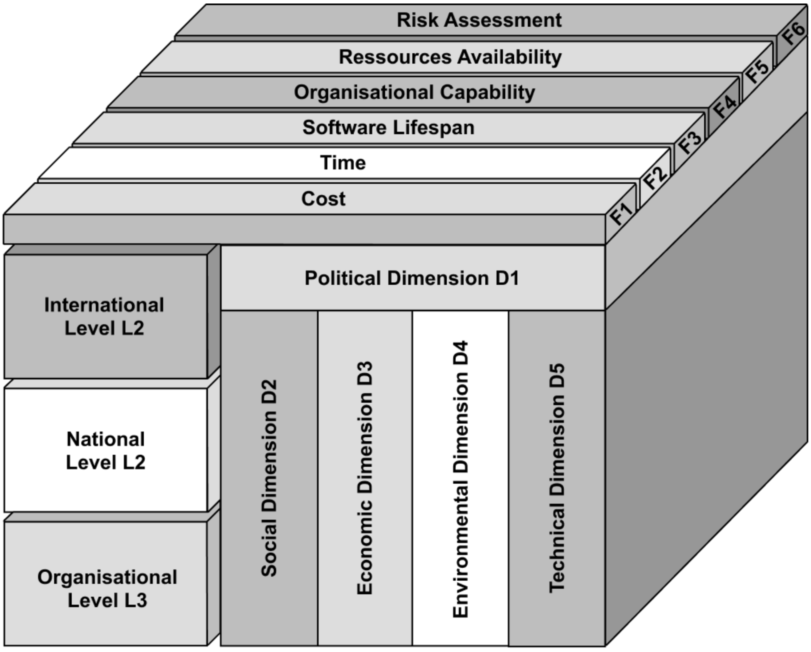
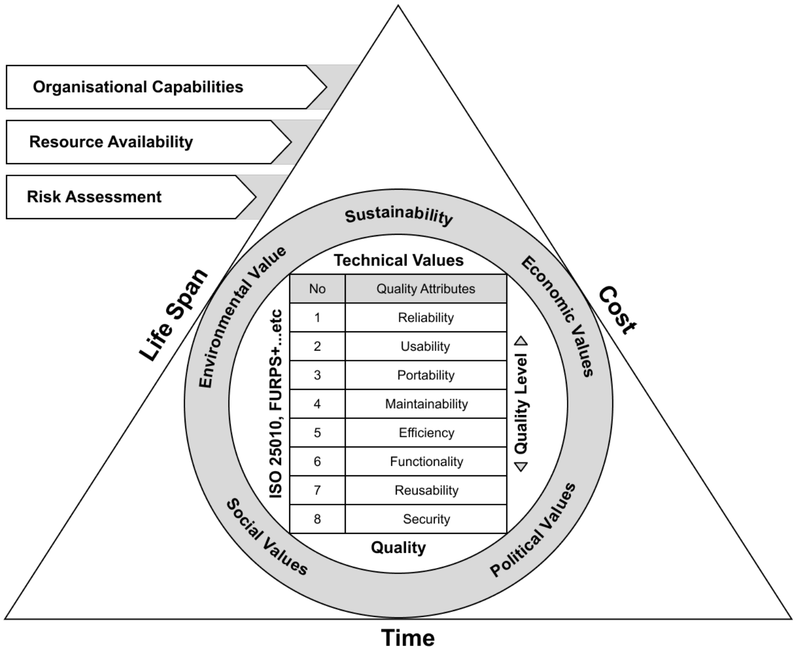


Figure 3.9: Conceptual framework of software sustainability (Aljarallah, 2021)

In addition, his approach includes six major constraints, labeled F1 through F6. He represents the relationships among these factors in an extended sustainability triangle for software (see Figure 3.10). This triangle is intended to provide a structured approach for making difficult decisions between various project requirements and sustainability aspects, with a strong emphasis on ensuring software quality is at its core.

Holistically, the framework offers a structured and practical approach for further unite the topics of sustainability and software. Attention is paid to the identification of necessary roles as well as clear, step-by-step instructions (seven steps in total) on how to apply the framework. Although this framework already integrates a good range of factors, Aljarallah himself criticizes the fact that only the design and development phase is considered during the model creation (Aljarallah, 2021).



No	Quality Attributes
1	Reliability
2	Usability
3	Portability
4	Maintainability
5	Efficiency
6	Functionality
7	Reusability
8	Security

Figure 3.10: Enhanced triangle for software sustainability (Aljarallah, 2021)

Sustainability Awareness Model: SusAD (Becker et al., 2015), SusAF (Duboc et al., 2020) and the BE-SusAF (Lammert et al., 2024a)

Despite many different approaches and models for integrating sustainability in software engineering, the research effort toward a practical and structured model is still ongoing. The best-known model to date, according to the number of publications, is the Sustainability Awareness Framework known as SusAF.

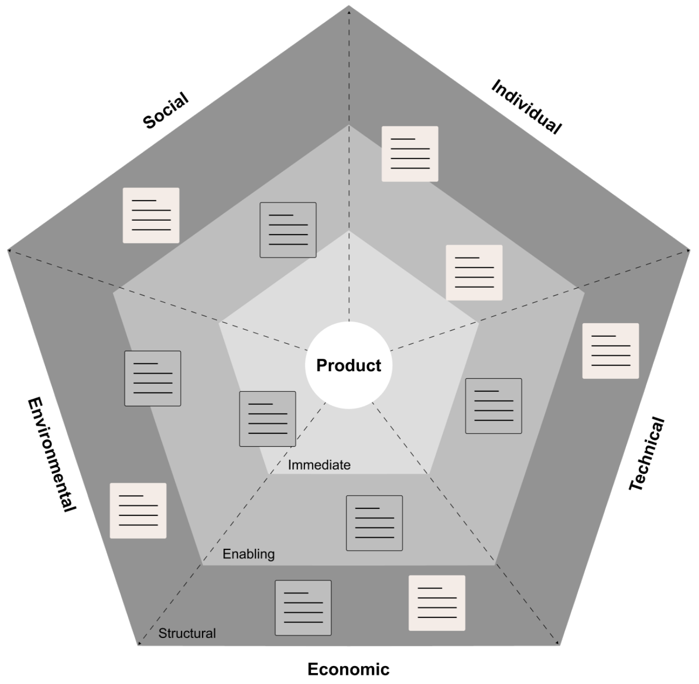


Figure 3.11: Sustainability awareness diagram, based on Becker et al. (2015) and Duboc et al. (2020)

This model was developed by a research group over several years and through several iterative loops. Some of the scientists who were involved in the creation of the Karlskrona Manifesto can also be found in the list of authors of the SusAF model. The core component of SusAF is a pentagon-shaped diagram, also known as the Sustainability Awareness Diagram, or SusAD for short. Figure 3.11 provides a visual representation of SusAD. Becker et al. developed SusAD to visualize and capture sustainability-related requirements throughout the software development process. This model applies the five-dimensional understanding of sustainability—economic, ecological, social, individual, and technical—while distinguishing between immediate, enabling, and structural impacts. The latter describes first-, second-, and third-order effects (Becker et al., 2015).

This framework's overarching goal is to raise awareness of sustainability impacts within the software domain. Accordingly, the SusAF covers the entire proposed model and, alongside the SusAD, comprises instructions and a questionnaire for each sustainability dimension. The topics queried for each dimension are listed in Table 3.1. Once responses to these questions have been collected through interviews, the framework captures the impacts and potential impact chains, visually mapping and linking them within the diagram (Duboc et al., 2020).

Table 3.1: Topics covered in the questionnaire sorted by sustainability dimensions, adapted from Duboc et al. (2020)

Dimension	Topic
Social	(1) Sense of community; (2) Trust; (3) Inclusiveness and diversity; (4) Equity; (5) Participation and communication
Individual	(1) Health; (2) Lifelong learning; (3) Privacy; (4) Safety; (5) Agency
Environmental	(1) Material and resources; (2) Soil, Atmospheric and water pollution; (3) Energy; (4) Biodiversity and land use; (5) Logistics and transportation
Economic	(1) Value; (2) Customer relationship management (CRM); (3) Supply chain; (4) Governance and processes; (5) Innovation and R&D
Technical	(1) Maintainability; (2) Usability; (3) Extensibility and adaptability; (4) Security; (5) Scalability

To transfer this theoretical version, explained by examples, completely into practice and thus use it for industrial case studies, Lammert proposes a business-oriented extension of the SusAF model, which is given the abbreviation BE-SusAF. This extension considers requirements derived from the industry and should, therefore, no longer be located directly by the software engineer but by the product manager. Figure 3.12

shows the artifact proposed by Lammert for the BE-SusAF model, which is divided into the preparation, workshop, and follow-up phases. Due to its relevance to this work, the addition of the Value Proposition Canvas (VPC), which is located directly before the implementation of the SusAF, and the addition of the modified BM canvas (TL-BMC) should be particularly emphasized. This adjustment reduces the previous strong focus on the design phase. It broadens it to include both the benefits of the software for the user (addressed through the VPC) and the business-oriented perspective (see TL-BMC). The BE-SusAF published in the article "The Business-Oriented Extension of the Sustainability Awareness Framework—A Design Science Study" (Lammert et al., 2024a) is a core component of his doctoral thesis defended in 2024 (Lammert, 2024). Due to the high thematic relevance in the industry, the collected findings have been transferred into a workbook that is accessible to a broad audience via the website www.suso.academy.

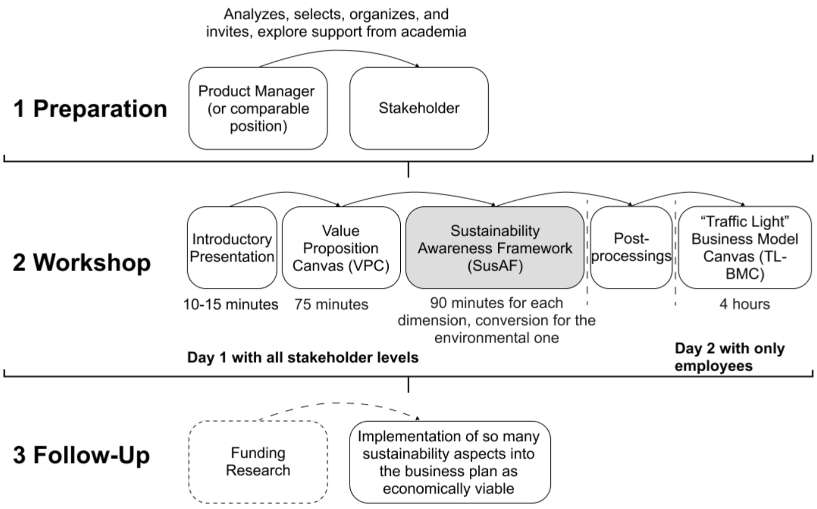

Analyzes, selects, organizes, and invites, explore support from academia
1 Preparation
Product Manager (or comparable position)
Stakeholder
2 Workshop
Introductory Presentation
Value Proposition Canvas (VPC)
Sustainability Awareness Framework (SusAF)
Post-processings
"Traffic Light" Business Model Canvas (TL-BMC)
10-15 minutes
75 minutes
90 minutes for each dimension, conversion for the environmental one
4 hours
Day 1 with all stakeholder levels
Day 2 with only employees
3 Follow-Up
Funding Research
Implementation of so many sustainability aspects into the business plan as economically viable

Figure 3.12: Business-oriented extension of SusAF (Lammert et al., 2024a)

While the extension of the framework offers valuable additions, the Sustainability Awareness Framework primarily focuses on an IT product's medium and long-term effects. Strategic and organizational framework conditions, which are crucial for successful integration after development and for the sustainable use of a software product, are not sufficiently addressed.

Given the current level of knowledge about sustainable software, this raises the question of whether it is productive and appropriate to discuss third-order impacts when no approach has yet been established in industry or practice for designing, developing, and effectively implementing new IT projects in a sustainable manner. To address this and other questions, a unified, practical, and easily understandable framework for sustainability in software projects is needed. From this perspective, this thesis considers all those software projects to be sustainable which fulfill the following three aspects:

First, software needs to promote a sustainable goal. Second, software applications are sustainable if they are used continuously and in the long term. Third, software is considered sustainable if it is successfully implemented in a designated environment. This objective depends on many different factors, which need to be defined according to the second research question (RQ2).

Following the presentation and discussion of the frameworks most closely aligned with the objectives of this study, the frameworks are evaluated based on the following seven criteria:

- *Key Dimensions of Sustainability*, meaning that all three dimensions of the triple bottom line approach are considered.
- *Multiple Lifecycle Phases,* meaning that the framework includes more phases than the design and development of software.
- *Strategic and Organizational Conditions* to ensure a holistic approach.
- *Technical Perspective,* as part of sustainability in software.

- *Non-Technical Perspective,* as part of sustainability in software.
- *Sustainability by Software,* meaning that the software has a sustainable purpose.
- *Practical Guidelines, ensuring that the methodology provide clear guidelines to apply in real-world scenarios.*

The assessment is divided into three levels: not, partially, and fully fulfilled. This approach ensures a clear and systematic comparison of the frameworks regarding each criterion, providing a comprehensive overview of their strengths and areas for improvement. Table 3.2 summarizes the evaluation results based on the previous explanations. Finally, it becomes clear that the proposed methodology should primarily address the need for strategic and organizational conditions as well as for sustainability by software, since the least investigation occurs here.

Table 3.2: Evaluation of related frameworks

Evaluation of the presented frameworks with regard to the chosen evaluation criteria **Question:** To what extent do the various frameworks address the key topics covered by the evaluation criteria? **Scale:** ● = fully ◐ = partially ○ = non		**Evaluation Criteria**						
					Sustainability in Software			
		Key Dimensions of Sustainability	**Multiple Lifecycle-Phases**	**Strategic and Organizational Conditions**	**Technical Perspective**	**Non-Technical Perspective**	**Sustainability by Software**	**Practical Guidelines**
Relevant Frameworks	**GreenSoft Model**	◐	●	○	◐	◐	◐	◐
	Generic Sustainable Software Star Model (GS3M)	●	○	◐	●	●	○	○
	Software Project Sustainability and Project Success (SPS Framework)	●	◐	◐	●	●	○	○
	Framework for Software Sustainability	●	◐	●	●	●	○	●
	Sustainability Awareness Model (SusAF, SusAD and BE-SusAF)	●	●	○	◐	◐	●	●
	Proposed Approach	●	●	●	●	●	●	●

3.4 Conclusion of Chapter 3

The chapter highlighted that, particularly over the past two to three years, an increasing number of practical models addressing sustainability in software engineering have emerged. In addition, active research communities have been observed for some time, contributing to a vibrant stream of research. The high relevance of the topic, coupled with the growing willingness of the professional community to develop new models and extended solution approaches, creates an ideal environment for active participation. The intensive and ongoing evaluation of published frameworks opens up the opportunity to strategically use the time until they are established to contribute further findings. In this way, the theoretical foundation can be further developed, and a substantial contribution to practical applicability can be made.

In this context, this chapter provided an overview of general re-engineering models and extensively explored the previous findings in sustainability in software engineering. The literature used for this purpose repeatedly showed gaps in a uniform definition of sustainability in software engineering, thereby complicating comparison with the presented models. Subchapter 3.3, therefore, exclusively tackled theoretical and practical models, considered important milestones for linking science and business in this thesis. The comparison of the nominated frameworks listed in Table 3.1 indicates a clear gap in the strategic and organizational integration of the software into the corporate context as well as in sustainability by software. This systematic consideration is based on the idea that although software can be developed in a technically "sustainable" way, it cannot be declared sustainable without a clear reference to the actual objectives of sustainability in terms of its ecological, economic, or social impact. The same applies to software applications without any connection to corporate strategy and, therefore, have no long-term relevance. Hence, software technologies for sustainable BMs imply that the software makes a long-term, value-adding, and resource-efficient contribution in the corporate context.

Existing methods do not fulfill the need for a strategic and pragmatic approach to meeting these challenges. **The identified research gap lies in developing a practice-oriented method that considers strategic, technical, and organizational aspects equally while incorporating sustainability by software.**

4 Methodology

This chapter represents the centerpiece of this dissertation, focusing on introducing a methodology for the strategic re-engineering of software technologies. It builds upon the theoretical foundations and state-of-the-art laid out in previous sections, offering a structured approach to master software technology, business objectives, and sustainability. In doing so, a notable gap in science and industry is closed, and theoretical insights and guidelines for practical implementation are provided.

The chapter is structured as follows: Section 4.1 focuses on the objective of the methodology, followed by Section 4.2, outlining the design of the methodology, including an overview of the different roles, the structure, and its process. Chapters 4.3, 4.4, and 4.5 present the key components of the methodology in detail, starting with the layer of purpose, through the layer of technology, to the layer of enablers. Finally, Section 4.6 provides a total evaluation template before Section 4.7 summarizes the key aspects of the methodology.

4.1 Objective of the Methodology

Following the title of this thesis, the primary objective of this methodology is to provide a well-structured approach tailored to position software for continuous and long-term success.

The concept of "strategic re-engineering" encompasses both a long-term vision and a holistic approach. It simultaneously aligns software development with broader organizational objectives, as outlined in the subtitle "in the value proposition of sustainable BMs."

Building on this foundation, the methodology in the following sections integrates two core transformations: digital and sustainable. DT is driven by software technology, which is the core component, while sustainable transformation acts as both the catalyst for the re-engineering process and the overarching goal the methodology seeks to achieve.

At this stage, and before introducing the methodology, it seems essential to clarify the term "software technology" to ensure consistency and a shared understanding. Therefore, the term is used interchangeably with "software application" or "software solution" and refers to the software as a finalized product ready to use by a human end user. Within the thesis, this approach applies exclusively to individual software projects.

In summary, the proposed methodology was conceived to respond to the need for a comprehensive planning and validation template for developing individual software applications within the context of sustainability.

4.2 Design of the Methodology

This chapter visualizes the design in a step-by-step approach to introduce the methodology in a structured and transparent way while ensuring that sufficient attention is given to details. It begins by presenting the key roles required throughout the entire re-engineering process, followed by focusing on the three main layers, the detailed architecture, and by guiding through the proposed process.

4.2.1 Roles

Many actors and roles are required to successfully merge the software, business, and sustainability aspects. During the creation of the methodology, nine leading roles were identified (see Figure 4.1). One role operates at the business–sustainability interface. Three roles are concentrated on promoting business aspects. One role handles the interface between business and software, while another concentrates on fulfilling the software part. In addition, the development team responsible for developing the software is assigned to the software–sustainability interface to meet the requirements from both sides. Finally, two roles are positioned in the nexus of all three sectors. The following section provides a detailed explanation of the roles depicted in Figure 4.1.

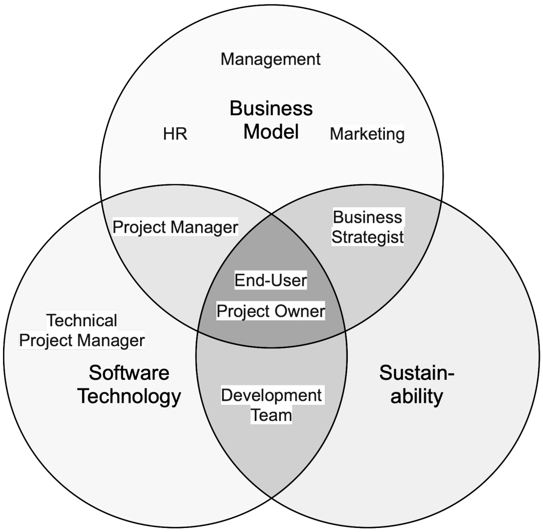


Figure 4.1: Key stakeholder roles for the methodology

Business Strategist

Within this methodology, the business strategist operates at the intersection of business and sustainability, focusing on providing strategic insights for long-term value creation. This role demonstrates a deep understanding of market dynamics. The main tasks are gathering requirements, conducting stakeholder analysis, modeling business processes, identifying improvements, analyzing data, and creating insightful reports for management. In a nutshell, it is about "translating strategy into action" (Paul and Tan, 2015) while collaborating with cross-functional teams. This role is critical in ensuring that sustainability is embedded in the overall business strategy.

Project Manager

From ideation to concept creation to the final go-live phase, a project manager manages transformation projects from the business perspective while staying aligned with the company´s strategy. They are crucial in orchestrating various interests and processes and ensuring smooth project management. In the given methodology, this role mainly focuses on delivering software projects on time, on budget, and with the expected quality. Besides the classical three project management dimensions (quality, cost, and time), a project manager takes additional care of the dimensions of sustainability and innovation. Leveraging digital tools and technologies, business project managers drive BM innovation.

Technical Project Manager

Technology experts handle the technical aspects of a new business opportunities. Their expertise lies in providing advice, designing solutions, and overseeing their implementation. This role is primarily responsible for the quality of the software application, including continuous process optimizations and further enhancements. Within the following methodology, the technical project manager is linked to the software aspect. Accordingly, this role supervises the frontend and backend development of a software application.

Development Team

The development team handles several roles. The roles that are required depend on both the selected development method and the type of software application being developed. Depending on the complexity and scope of the project, several team members of the same function are often required. Core roles in a software development team are the software architect, who designs the entire system architecture, and the frontend and backend developer. A frontend developer builds the software user interface that the end user faces and interacts with. The frontend is, therefore, the interface of a digital application that connects people with technology. A frontend developer works very closely with the user interface (UI) and user experience (UX) designers, who initially define and design the look and feel of the software user interface. Moreover, behind each surface is a backend environment responsible for the functionality of buttons, interrelated connections, and APIs. This means this role develops and maintains the server-side logic of software systems, taking responsibility for setting up a seamless exchange between frontend and server/database. Optionally, the role of full-stack developer is available. Developers capable of handling frontend and backend environments are called full-stack developers. This role is characterized as versatile and highly capable of handling complexity and different layers. Full-stack developers are valuable assets within a software team as they are very flexible and provide a holistic understanding of the whole underlying software technology. Alongside the main functions, other important roles abound, such as software quality assurance (SQA) and DevOps engineers. SQA conducts manual and automatic tests of the software functions to remove mistakes, known as software bugs, while DevOps engineers take over administrative tasks, among others.

Management

As part of the top executive level, management plays a crucial role in making strategic decisions that shape the overall direction of a company. This methodology emphasizes that management's primary task is to foster an open and innovative work culture where changes are embraced in response to market developments. By cultivating such an environment, management enables the organization to adapt and thrive in a dynamic business landscape.

Product Owner

Product owners are responsible for the overall outcome, impact, and success of the software project and are therefore assigned to the intersection of all three dimensions. While the aforementioned project managers are involved in the day-to-day business and project delivery, product owners monitor the project from a bigger perspective, focusing on the software value proposition. Therefore, product owners take care of collecting user requirements, translating them into product features, and prioritizing them for development (Sommerville, 2016). In addition, a product owner monitors communication with different stakeholders who are involved in the software project to ensure transparency and collaboration throughout the development process.

Marketing

In this methodology, the marketing role includes the task of communicating the implemented software project effectively and efficiently, both internally and externally. The main task here is to develop communication materials that serve to incorporate software technology into day-to-day operations as quickly as possible. For example, it is often necessary to provide information about the software application's benefits, as well as training on how to apply and use the software. A well-thought-out marketing strategy that picks up all stakeholders as early as possible influences the success of a project, highlighting the importance of this role.

HR

In addition to marketing, HR is equally important, as this role also impacts an organization's acceptance of innovation and changes. One of the main factors to consider here is the influence on a positive corporate culture and readiness for change within the organization.

End User

In this methodology, customers do not take an active role but serves as a target reference. The goal is to understand their needs and improve their services. The customer's designation, however, focuses not only on the customer intending to buy but on the general end user for whom the elaborated solution approach is intended. For example, this can also be an employee. In summary, the customer is the target person who will use the software technology.

4.2.2 Building Blocks and Layers

The general re-engineering process, as outlined in Chapter 3.1, involves the three steps of analyzing, designing, and implementing an existing software system, with the aim of improving or adapting the current system.

When applying this re-engineering process to software in the nominated area of interest, the value proposition in sustainable BMs, this thesis assigns the analysis phase to the task of the software, the design phase to the software itself, referred to as the tool, and the implementation phase to human factors. These assigned components are referred to hereafter as building blocks, as visualized in the middle part of Figure 4.2.

To meet this holistic perspective, all three building blocks must be addressed collectively and with equal importance. Therefore, this work argues for a holistic approach in which the harmonization of tasks, tools, and humans is essential to develop software for sustainable BMs. Furthermore, this methodology is based on the idea that software for sustainable BMs should not only fulfill a sustainable task but should also carry the idea of sustainability within itself. In summary, the concept is based on the three building blocks, which include the value resulting from the task of the software, the software itself as a functional tool, and the human factor that influences the interactivity and utility of the system.

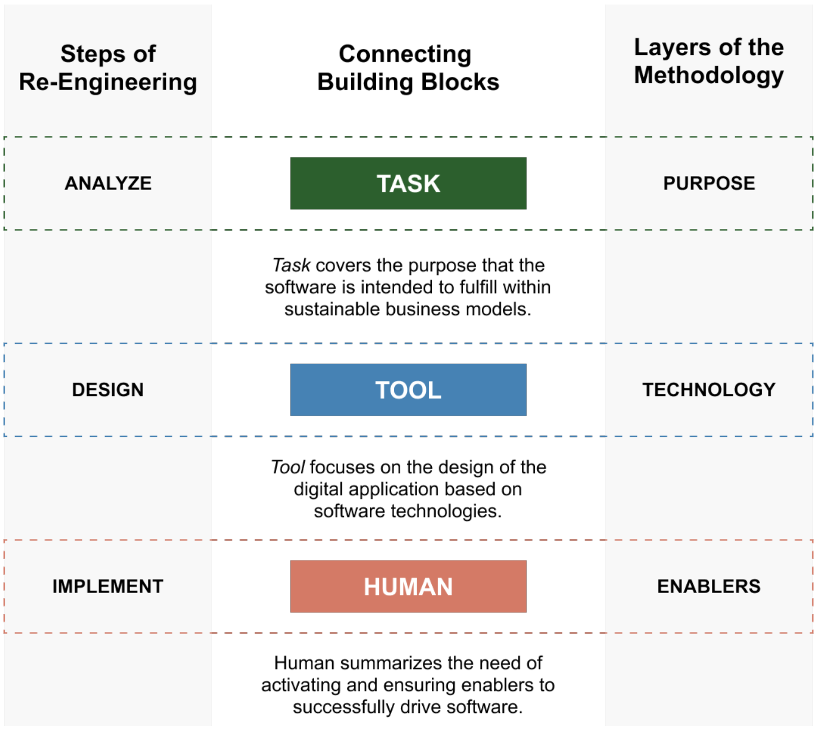

Steps of Re-Engineering
Connecting Building Blocks
Layers of the Methodology
ANALYZE
TASK
PURPOSE
Task covers the purpose that the software is intended to fulfill within sustainable business models.
DESIGN
TOOL
TECHNOLOGY
Tool focuses on the design of the digital application based on software technologies.
IMPLEMENT
HUMAN
ENABLERS
Human summarizes the need of activating and ensuring enablers to successfully drive software.

Figure 4.2: Derivation of the methodology structure

The three layers of the methodology were determined by referring to the knowledge gained in the theoretical foundations, enriched by the practical experience gained in software development and finally by coupling the classic re-engineering steps—analysis, design, and implementation—with the defined building blocks: task, tool, and human. This leads to the identification of the layers: Software Purpose, Software Technology, and Software Enablers, visualized in Figure 4.2. The building block 'task' involves analyzing the purpose of the software. The building block ‘tool’ focuses on determining the design of the software technology. Finally, the building block ‘human’ addresses the implementation of software enablers.

In brief, this thesis defines the strategic re-engineering of software concerning sustainable BMs as a **coordinated interplay between the iterative layers of purpose, technology, and enablers of the software.**

4.2.3 Architecture

The architecture of the proposed methodology is structured around the three previously defined layers, which are characterized by an implementation-oriented approach and an iterative procedure. Figure 4.3 illustrates the architecture of the methodology, which includes seven segments: two layers with two segments and one layer with three segments.

The first layer of the methodology focuses on defining the software's purpose. Architecturally, this layer is centrally positioned, as it serves as the core to which all other components contribute. This section is critical in articulating the primary objective that the software is intended to address. Specifically, it emphasizes the analysis of digital and sustainable transformations. From the DT perspective, identifying a digital business purpose is essential, while from the sustainability transformation viewpoint, a corresponding sustainable purpose must be addressed. At this point, emphasizing the recent academic recognition that defining a sustainability purpose for software enhances the software development process is crucial (Noman et al., 2024; Souza et al., 2024).

To ensure that both digital business and sustainability aspects are adequately addressed, two impact segments are incorporated as foundational elements. As illustrated in Figure 4.3, the business impact segment defines the rationale behind initiating a DT project, while the sustainable impact segment underscores the software application's contribution to sustainability goals.

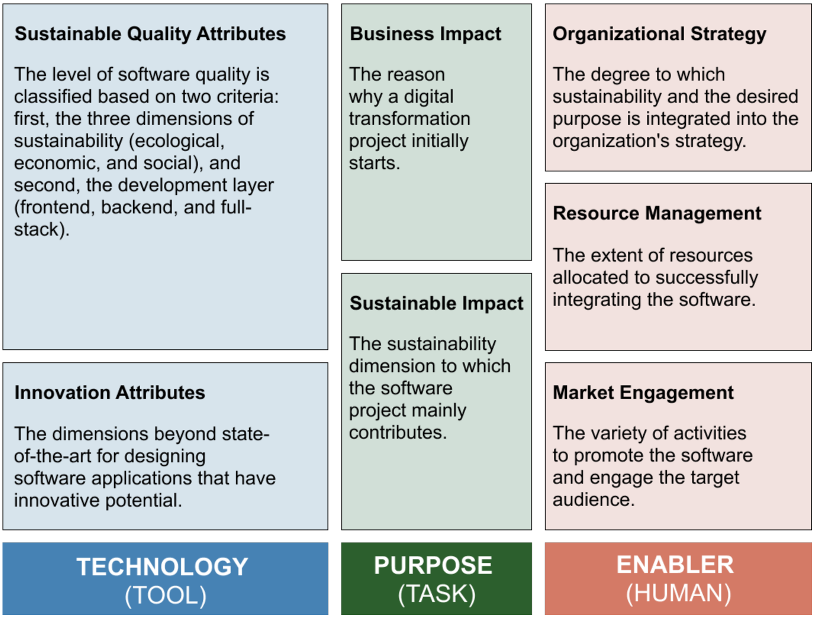

Sustainable Quality Attributes
The level of software quality is classified based on two criteria: first, the three dimensions of sustainability (ecological, economic, and social), and second, the development layer (frontend, backend, and full-stack).
Innovation Attributes
The dimensions beyond state-of-the-art for designing software applications that have innovative potential.
Business Impact
The reason why a digital transformation project initially starts.
Sustainable Impact
The sustainability dimension to which the software project mainly contributes.
Organizational Strategy
The degree to which sustainability and the desired purpose is integrated into the organization's strategy.
Resource Management
The extent of resources allocated to successfully integrating the software.
Market Engagement
The variety of activities to promote the software and engage the target audience.
TECHNOLOGY (TOOL)
PURPOSE (TASK)
ENABLER (HUMAN)

Figure 4.3: Architecture of the methodology

The second layer centers on the design of the software technology and is placed on the left side from the software purpose. To address how software technology must be designed to fulfill sustainable BMs, this layer includes two core segments: sustainable quality attributes and innovation attributes. Accordingly, the upper segment, sustainable quality attributes, identifies the degree of software quality based on non-functional requirements. The nine identified requirements are thoroughly examined in Section 4.4 and categorized based on their sustainable orientation and technological development level to facilitate the practical consideration of non-functional requirements for software technology.

The lower segment, without any hierarchical distinction or prioritization relative to other segments, integrates the aspect of innovation into software technology design. Innovation, as a constant driver of change and transformation, ensures the creation of future-oriented designs within the framework of this methodology. Accordingly, the embedded innovation attributes serve to ensure customer satisfaction and the associated long-term and, therefore, sustainable utilization by considering cutting-edge technologies and state-of-the-art design elements.

The third layer of the methodology introduces the concept of software enablers, which are pivotal in facilitating the implementation and success of the software's purpose. Positioned as a critical support structure, this layer ensures that digital and sustainable impacts can be effectively realized. The layer of software enablers is divided into three distinct segments, each playing a vital role in driving the overall performance of software technology: organizational strategy, resource management, and market engagement.

The first segment, organizational strategy, focuses on aligning the company's governance and culture with digitalization and sustainability goals, shaping the software re-engineering environment. As a further segment within this layer, resource management focuses on the extent of different types of resources dedicated to successfully integrating software technology. Finally, the market engagement segment emphasizes the importance of customer success initiatives, outlining activities that actively engage the target audience and promote the software's adoption and long-term impact.

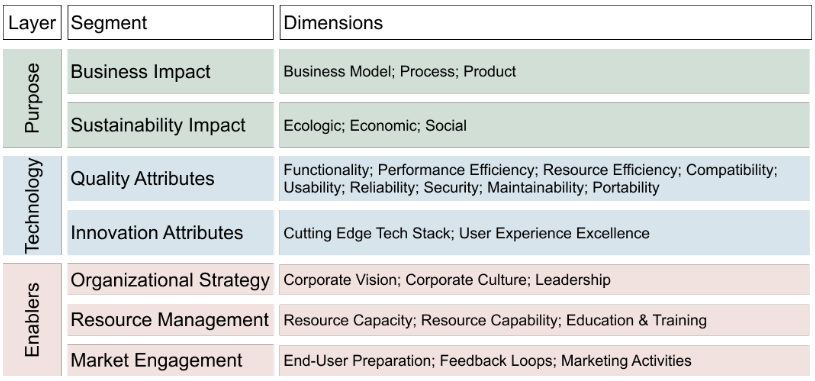
The overall visualization of the methodology architecture (see Figure 4.3) is inspired by Osterwalder's BM canvas. It is important to note that the comparison with the BM canvas (introduced in Chapter 2.3.1) is limited to the structure and visual representation of the methodology and was chosen due to its frequent application in the information systems (IS) discipline for analyzing technology use in business contexts (Steininger, 2018). Furthermore, this approach aims to leverage the widespread familiarity and acceptance of the canvas in practice, facilitating quicker adoption of the proposed methodology through existing and established user experience. Therefore, the center highlights the purpose of the software, which can be compared to the value proposition of BMs. The left part focuses on the design of software technology, which matches the BM value creation part. In contrast, the right part takes care of the activation of key enablers to ensure successful implementation, which illustrates the BM value delivery part.

4.2.4 Process

As previously mentioned, the methodology is particularly relevant for developing custom software. It can be applied in two specific scenarios. First, it can be used to evaluate already developed and/or implemented custom software, aiming to review the current status. Second, it applies when developing a new software application on a custom basis. In this case, the methodology serves as an excellent tool for assessing whether all relevant aspects have been considered or for identifying which layers require further improvement to eliminate potential obstacles at an early stage.

Following the identification of the relevant stakeholders and the presentation of the building blocks, layers, and overall architecture of the methodology in the previous sub-chapters, this section focuses on outlining the procedure. This procedure should not be viewed as rigid but as a guideline for best understanding the methodology. Notably, the methodology presented in Chapter 4 is not intended as a replacement for existing software development models but rather as an extension. Its purpose is to facilitate the targeted and successful development and integration of software within the company under the umbrella of sustainability.

To introduce the methodology's process, Figure 4.4 begins by breaking down each segment into multiple dimensions, aiming to provide a clearer overview of what each layer encompasses. We clarify beforehand that Figure 4.5 illustrates how the identified roles impact each segment and outlines their specific areas of responsibility. Thereby, the product owner has the lead within the project and therefore a general responsibility across all layers and segments, as this role oversees alignment between strategic objectives, operational execution, and stakeholder interests. Figure 4.6 focuses on the step-by-step process of the methodology, introducing the associated toolkit.



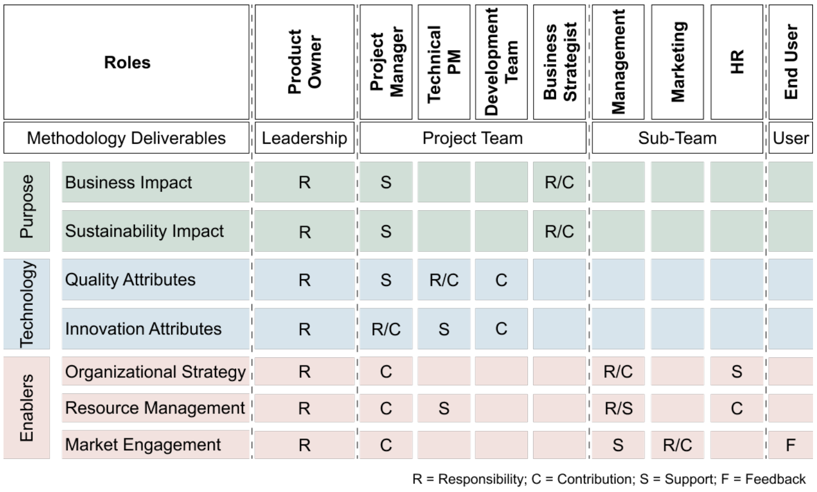
Layer	Segment	Dimensions
Purpose	Business Impact	Business Model; Process; Product
	Sustainability Impact	Ecologic; Economic; Social
Technology	Quality Attributes	Functionality; Performance Efficiency; Resource Efficiency; Compatibility; Usability; Reliability; Security; Maintainability; Portability
	Innovation Attributes	Cutting Edge Tech Stack; User Experience Excellence
Enablers	Organizational Strategy	Corporate Vision; Corporate Culture; Leadership
	Resource Management	Resource Capacity; Resource Capability; Education & Training
	Market Engagement	End-User Preparation; Feedback Loops; Marketing Activities

Figure 4.4: Dimensions of each methodology segment

The recommended procedure starts with the business impact segment and defines the digital value proposition of the custom software application. This definition pertains to either a digital BM, the digitalization of a process, or a digital product. In practice, the responsibility for this task falls within the business strategist, with support from the project manager. To facilitate implementation, this thesis provides templates with fill-in-the-blank sections for practical use.

Subsequently, attention shifts to defining the software's sustainable value proposition, which may encompass one or multiple dimensions of sustainability, as depicted in the sustainable impact segment of Figure

4.4. Similar to the previous step, the business strategist primarily undertakes this task, and a template is included as part of the methodology to assist with the practical execution application.



Roles		Product Owner	Project Manager	Technical PM	Development Team	Business Strategist	Management	Marketing	HR	End User
Methodology Deliverables		Leadership	Project Team				Sub-Team			User
Purpose	Business Impact	R	S			R/C				
	Sustainability Impact	R	S			R/C				
Technology	Quality Attributes	R	S	R/C	C					
	Innovation Attributes	R	R/C	S	C					
Enablers	Organizational Strategy	R	C				R/C		S	
	Resource Management	R	C	S			R/S		C	
	Market Engagement	R	C				S	R/C		F

R = Responsibility; C = Contribution; S = Support; F = Feedback

Figure 4.5: Roles and responsibilities

Step two focuses on analyzing and evaluating the nine sustainable quality attributes. A unique model is employed to conduct the evaluation within a multidimensional context. This evaluation is assigned to the technical project manager, while the responsible developers—whether frontend, backend, or full-stack developers—contribute by executing the implementation.

To complete the technology layer, this step also includes a revision of the innovation attributes. This step mainly examines the potential for utilizing cutting-edge technologies and approaches that strive for excellence in the user experience. The responsibility for this evaluation lies with project managers, supported by technical project managers. As Figure 4.6 shows, the entire technology layer relies on models for practical implementation, whereas the Kano model will be discussed in greater detail regarding innovation attributes.

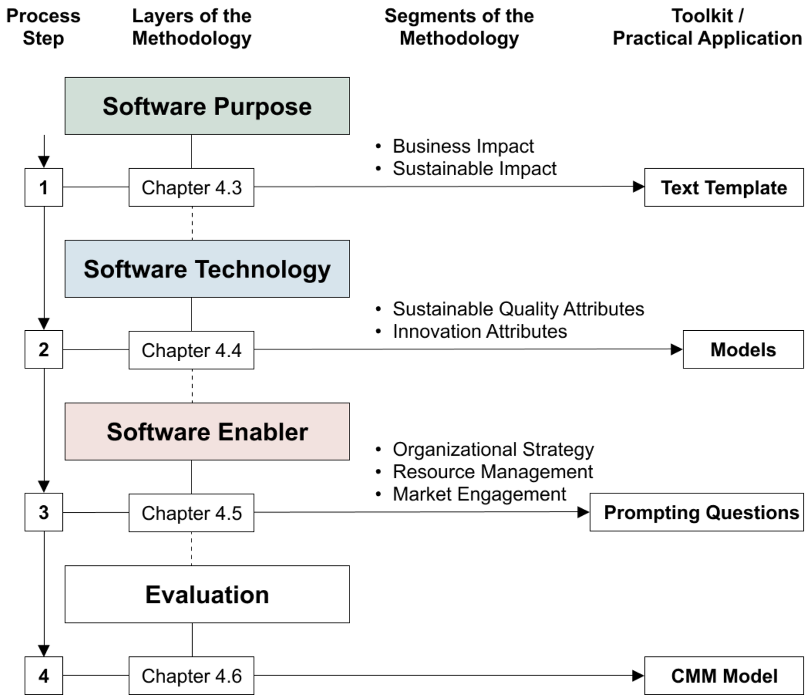


Figure 4.6: Process of the methodology

Finally, the reader should be directed to the right side of the predefined architecture. At the conclusion of each segment, prompting questions are provided as a consistent, practical tool. The segment on organizational strategy delves into three key dimensions: corporate vision, leadership, and organizational culture. The management is responsible for these dimensions while the HR department strongly supports for execution.

The resource management segment also comprises three dimensions: resource capacity, resource capability, and resource education and training. The product owner and management primarily oversee this segment, while HR serves as a key contributor.

Like each of the three enabler segments, the seventh segment, market engagement, is divided into three additional dimensions. In this case, the focus is on end-user support, continuous feedback loops, and marketing activities. As with the previous segment, the project owner is primarily responsible for the former dimension, while the marketing department is mainly accountable for the latter.

The last step of the methodology is presented in Chapter 4.6 with the introduction of an evaluation template. The template refers to the CMM model already presented in the theoretical foundation section and is designed to gain insights into the maturity level of each segment and to derive suitable improvement measures.

As Figure 4.6 clearly illustrates, the following three subchapters are organized according to the three layers of the methodology. Each subchapter focuses mainly on the function of its respective segments and provides a detailed explanation of the individual dimensions within those segments.

4.3 Layer of Software Purpose

Nowadays, digitalization and sustainability are the key drivers for innovative projects with transformative characteristics. Transformational processes always start with the question "Why?" in detail, why is change necessary, and what is the expected outcome of the innovative and transformative initiative? This thesis focuses on the two mentioned drivers: digitalization and sustainability. First, defining which aspect of DT should be addressed is crucial. Therefore, Chapter 4.3.1 emphasizes the three main types of applying digital technologies in an innovative manner, starting with BM innovation through process to product innovation. The aim is to define the DT part of the software´s value proposition. To further refine the purpose of the software technology and to adapt to the second transformation, sustainability, Chapter 4.3.2 focuses on the sustainability aspect of the value proposition. Therefore, the aim is to define the sustainable impact of re-engineered software.

Unlike the original principle of the triple bottom line, which aims to address all three sustainability dimensions equally, the proposed methodology is based on the fundamental idea that a DT project does not have to address all three dimensions simultaneously. Instead, it is more target-oriented to focus on one specific aspect and pursue it efficiently. To summarize, this subchapter aims to describe the first layer of the methodology, which is about defining the digital and sustainable impact of the underlying software technology.

4.3.1 Digital Business Impact

Digitalization is a cornerstone of innovation and transformation in today's rapidly evolving business environment. Changes in a company’s BM, services, products, or organizational and governance structures are crucial for staying competitive (Hess et al., 2016). The information system literature highlights the importance of digitalizing products, processes, or organizational structures (Silvi et al., 2023; Weill and Woerner, 2013). In contrast, the strategy and innovation management literature emphasizes the enhancement, automation, or creation of entirely new processes and activities (Piepponen et al., 2022).

In this context, adopting the DT aspect into a software's value proposition involves three distinct innovation areas: processes, products, and the BM of a company. Figure 4.7 shows these three levels of innovation by classifying them according to their value to society and business. Thus, process innovation creates less value, whereas BM innovation provides the greatest value on both axes.

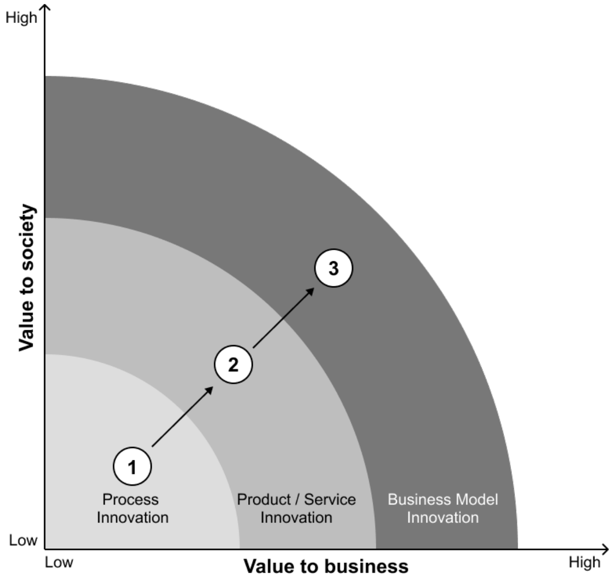


Figure 4.7: Progression of innovation levels, adapted from Clinton and Whisnant (2019)

Over a five-year period, BM innovation proved to be 6% more profitable than product or process innovation alone (Gassmann et al., 2017). By exploring these facets, this chapter defines the DT component, also referred to as business impact, of the software's value proposition by highlighting that each specific area has its own purpose, benefits, and challenges. Therefore, the digital business value proposition is defined as a set of benefits related to the target areas that the software offers to its users.

Process

Software is key in process innovation within or outside a company. A process involves a structured set of steps to achieve a task (Abijith et al., 2013). A classic definition of processes within a business context defines it as "a collection of activities that takes one or more kinds of input and creates an output that is of value to the customer”(Hammer and Champy, 1993). Typical processes in mechanical engineering companies include development, production planning, manufacturing, logistics, supply chain management, customer support, maintenance, HR administration, procurement, ordering, and invoicing. Business processes can be categorized into two types: operational and management processes (Abijith et al., 2013). The processes in a company are diverse, often of a complex nature, and require careful coordination and management to ensure efficiency and alignment with organizational goals. Linked to the software purpose, this section highlights typical digitization, digitalization, and DT process projects (see Figure 4.8). These types are used to innovate analog processes in the company in order to stay competitive or to further improve digital processes by applying cutting-edge technologies like robotic process automation to create greater value. Another use case involves adapting an existing process to changing customer needs and requirements. Proven benefits of a digital process transformation project include reduced costs and errors, increased efficiency as well as transparency and traceability through the availability of data. Software utilization to map digital processes enables companies to collect data and information (regarding machines, user behavior, etc.), create new customer interfaces, and automate processes. This creates a more dynamic environment with greater scalability, individualization, and flexibility along with enhanced communication and collaboration.

Two examples are given to illustrate the impact of software on process digitalization and highlight its applied purposes. First, chatbots and AI-driven software systems provide 24/7 service in customer support, improving the user experience with immediate responses and personalized support based on data analysis. Second, Siemens exemplifies software utilization in manufacturing processes. Their digital factory has increased productivity and reduced error rates by utilizing digital technologies such as IoT, automation, and big data analytics. Real-time monitoring, digital twins, and predictive maintenance optimize maintenance cycles and minimize unplanned downtimes (Siemens, 2024). These examples demonstrate software's pivotal role in improving operational efficiency in business processes.

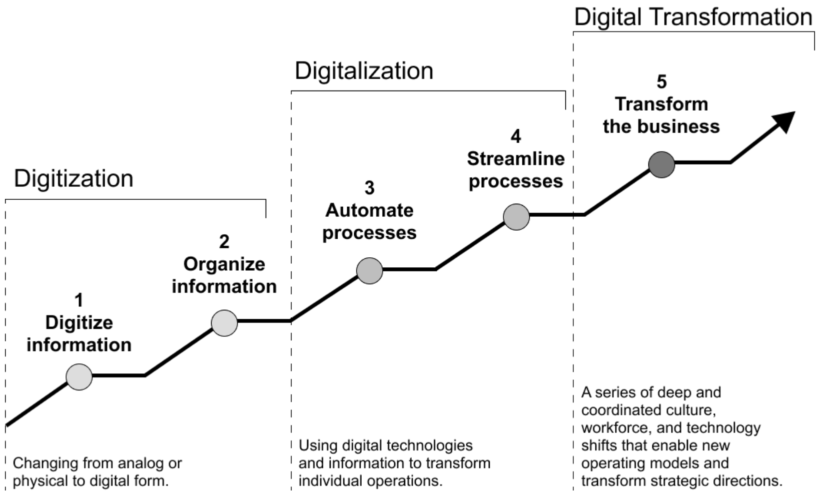

Digital Transformation
Digitalization
Digitization
1
Digitize information
2
Organize information
3
Automate processes
4
Streamline processes
5
Transform the business
Changing from analog or physical to digital form.
Using digital technologies and information to transform individual operations.
A series of deep and coordinated culture, workforce, and technology shifts that enable new operating models and transform strategic directions.

Figure 4.8: Difference between digitization, digitalization, and digital transformation, adapted from Reinitz (2020)

Software technologies commonly used in process digitalization include Microsoft Power Automate, SAP, ServiceNow, and Salesforce. Exemplary, the first one focuses on task automation and workflow management, while Salesforce excels in CRM with customizable marketing tools. The integration options include diverse APIs, ensuring compatibility with various systems. When custom solutions are applied in practice, they often require or benefit from a functional interface (API) that connects to one of the listed technologies, especially if it is already in use within the organization.

Product and/or Service

Alongside BM and process innovation, software technologies are frequently used for product innovation. As Heany noted in 1983, the range and the associated risk of product innovations is broad. Product innovation begins with a change of style in an established product and extends to entirely new products that address new markets and industries. Software technologies can thus generate product improvements, new products for existing markets, and new products for new markets (Heany, 1983). These types of products are referred to as digital products. Digital products, such as digital infotainment systems in automobiles, are defined as "products or services that are either embodied in information and communication technologies or enabled by them" (Lyytinen et al., 2015). Thus, in case where the purpose of software implementation is to provide a product, the outcome is often characterized by attributes such as scalability, accessibility, reproducibility, interactivity, and an immaterial nature. Moreover, digital products can be customized to meet the needs and preferences of users. The primary aim of digital products or services is to enhance customer retention by offering a high degree of usability along with carefully designed aesthetic features and engagement mechanisms (Nylén and Holmström, 2014). To successfully and quickly launch digital products in the company, among customers and on the market, linking them to a company's core business by, for example, adding digital components to traditional products is useful. Another strategy to effectively launch digital products with an impact is establishing closer connections with customers or users. For instance, utilizing a product configurator as a software tool can markedly enhance this strategy. A product configurator allows customers to tailor products to their specific preferences and requirements, offering interactive visualizations, such as 3D models, and operating within predefined rules to ensure feasibility and manufacturability. This approach not only enhances the customer experience by providing personalized options but also drives increased sales, generates valuable data-driven insights, and offers competitive benefits on the market.

Business Model

In traditional research, BMs and their value propositions are usually analyzed in the context of how they are transformed and enhanced through the application of software technologies. This means that digital, in general, or software technologies, in particular, enable companies to create new BMs, including cutting-edge technologies like blockchain, IoT, (generative) AI, or virtual and augmented realities (Böttcher and Krcmar, 2023). However, the proposed approach reverses the focus by placing the software in the central role. The software itself is described as embodying a value proposition that markedly influences and shapes a company´s BM. As defined in Chapter 2.2, a BM describes how a company creates, delivers, and captures value (Osterwalder, 2004; Teece, 2010). This perspective emphasizes the software's primary contributions, demonstrating how its capabilities and functionalities drive BM evolution and effectiveness, rather than merely supporting it. This shift in focus underscores the crucial role of software in enabling innovative and digital BMs.

A prime example of digital technologies creating new value propositions is the software developed by Netflix. This software transitioned the company's BM from renting physical media to offering a digital streaming service (Vial, 2019). Specifically, the software provides a seamless and user-friendly platform for streaming digital content. It manages a vast media library and ensures high-quality, reliable content delivery. The software also implements robust search and recommendation algorithms to enhance user experience, supports secure and scalable payment and subscription systems, and gathers and analyzes user data to continuously improve and personalize the service.

This example clearly illustrates that software technology facilitates the establishment of new BMs to remain competitive in the evolving market (Chesbrough, 2010; Teece, 2010). However, it also demonstrates that the success of a BM is not solely contingent on the software (Böttcher and Krcmar, 2023). Even the most user-friendly, functional, and secure software is ineffective if the underlying concept of the digital BM has not been fully conceptualized and validated. Mastering the balancing act between an individual characteristic and a holistic approach requires understanding the strategic purpose of the software and its BM objective.

Practical Guide: 'Text Template for Business Impact'

This methodology offers a clear, standardized template for defining the digital business value proposition by articulating the purpose of the software and identifying key elements such as:

- Target User Group
- Objective
- User Needs
- Functions and Features of the Software
- Benefits and Advantages
- Outcomes and Goals

Template "Digital Business Value Proposition":

The digital business value proposition of [Software Application] is to empower [Target Users] by seamlessly implementing a new digital [Business Model / Process / Product] that [Objectives]. Through its [Functions and Features], the software addresses [User Needs], resulting in [Benefits]. Unlike [Competitor], the software offers [Advantages], enabling [Outcomes] such as [Examples of Positive Outcomes]. This positions [Software Application] to achieve [Strategic Goals] more effectively and efficiently.

4.3.2 Sustainable Impact

Similar to the digital business value proposition, the second part of the first step of the proposed methodology concentrates on defining the sustainable value proposition of the software system. Adding a dedicated sustainability dimension to the software purpose empowers companies to gain operational improvements, opportunities for innovation, and strategic growth while accelerating their competitive advantage (Dao et al., 2011). In doing so, longevity and the achievement of the strategic re-engineering of software technologies within sustainable BMs can be ensured. In research, the sustainable value proposition is defined as "a promise on the economic, environmental and social benefits that a firm's offering delivers to customers and society at large, considering both short-term profits and long-term sustainability" (Patala et al., 2016). Hence, the sustainable value proposition reflects the centerpiece of sustainability by software, which in turn refers to software utilization to create a positive impact on the planet, economy, and people.

In detail and within this methodology, the sustainable value proposition of software refers to the three dimensions of sustainability while focusing on one dimension that will be most positively influenced by software implementation and usage. This means that not all three dimensions of sustainability need to be addressed equally. Instead, this work recommends focusing on one dimension of the triple bottom line approach, while multiple concentrations are possible and reciprocal interactions are known. Figure 4.9 provides an overview of possible sustainable aspects that can be addressed through software usage.

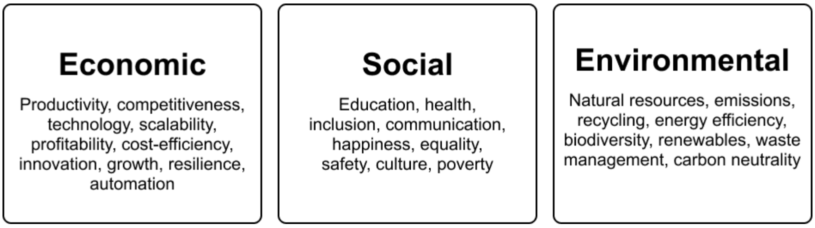

Economic
Productivity, competitiveness, technology, scalability, profitability, cost-efficiency, innovation, growth, resilience, automation
Social
Education, health, inclusion, communication, happiness, equality, safety, culture, poverty
Environmental
Natural resources, emissions, recycling, energy efficiency, biodiversity, renewables, waste management, carbon neutrality

Figure 4.9: Sustainable aspects

The one-dimensional approach is emphasized in order to develop a transparent and more focused purpose of the software, which is more feasible to execute and quantify and to achieve specific sustainable goals more efficiently. By defining the sustainable value proposition, the software can effectively communicate its value toward one or more dimensions and foster its commitment to sustainability.

Economical Aspect

A software application designed and developed to enhance the economic layer of sustainability embodies growth, innovation, new market development, quality/cost/time improvements, and resilience. Consequently, generating economic value through software technologies leads to cost savings, higher profit, increased return on investment, and long-term viability (Evans et al., 2017; Vladimirova, 2019). It aims to enable new BMs and expand business opportunities while ensuring operational stability and long-term risk reduction. Furthermore, this software type enhances a company's reputation and brand value, contributing to its attractiveness and market competitiveness, thereby boosting overall economic well-being through increased employment (Laukkanen and Tura, 2020). The software is also economically oriented if, for example, the application can drive innovation, optimize the allocation of resources or increase productivity. An example of economic benefits can be found in numerous software solutions that utilize the latest technologies to make standardized processes more efficient.

This is often accompanied by time savings, gaining of insights as well as time and space-independent usage.

Social Aspect

Socially oriented software technology applications include factors such as health, safety, empowerment, accessibility, and quality of life. Additionally, any software applications designed to implement and enhance the principles of diversity, equity, and inclusion (DEI) within organizations are also categorized under the 'social' dimension.

The benefits of software applications with a social focus, both internal and external from an organizational perspective, are extensive. For instance, implementing a software solution for automating internal documentation processes can enhance employee motivation and performance, leading to increased overall job satisfaction. Similarly, employing a software system for virtual training can ensure employee safety and improve working and training conditions, particularly in high-risk work environments (Patala et al., 2016).

Following the principle of this methodology, software applications with a social value proposition also tackle topics relating to law and regulations, respect and ethical principles, and increasing social well-being toward happiness and cohesion (Laukkanen and Tura, 2020). Another important point, elaborated in detail by Moises De Souza, is the role of software applications in promoting the social aspects of community and belonging. These applications can foster stronger connections among employees, facilitate collaboration, and create a sense of inclusion within the organization. By empowering users to exchange and interact through social channels, such digital platforms can help build a supportive work environment where employees feel valued and connected. This sense of community enhances job satisfaction and contributes to higher retention rates and a more cohesive organizational culture (Moises De Souza et al., 2024).

Environmental Aspect

Resources, waste, energy, and emissions must be reduced to fulfill the ecological objective of protecting the planet. Information technologies, especially software technologies, can play a key role in **reducing negative** ecological impacts and **creating positive** outcomes for the environment. Reducing negative impacts can be addressed by enhancing resource efficiency, preventing pollution and waste, decreasing energy use, and saving materials. Creating a positive impact on the planet can be achieved by promoting a circular economy, responsible usage, and sustainable consumption (Evans et al., 2017; Laukkanen and Tura, 2022, 2020; Vladimirova, 2019).

Rosário and Dias (2022) demonstrated that digital technologies markedly enhance sustainability by reducing waste and carbon emissions. They highlighted that digital solutions can be used to gain information on how negative climate impacts can be measured, controlled, and managed. They also added that datasets are the foundation for promoting and making positive, sustainable decisions. Furthermore, the 6R approach is a model primarily suited for manufacturing companies to develop digital products, processes, or BMs, or to reconsider their sustainable orientation. The 6R approach is known industry-wide and represents further development of the 3R approach, which originally covers the principles of reduce, reuse, and recycle. The intention of the software developed is to reduce and reuse the consumption of materials and resources through data analysis and automation and to support and empower the process of recycling natural resources to extend their life cycle (Zhang et al., 2013). The described 3R approach can be extended to the 6R approach by adding rethink, refuse, and repair or recover, remanufacture, and redesign (Jayal et al., 2010). Figure 4.10 provides further insights and shows one version of the extended 9R strategy. The 9R strategies should be considered from the perspective of how software can play a pivotal role in achieving these dimensions, such as enabling recycling through digital platforms, optimizing repair and refurbishment processes with predictive maintenance tools, or facilitating reuse by connecting users through shared digital marketplaces.

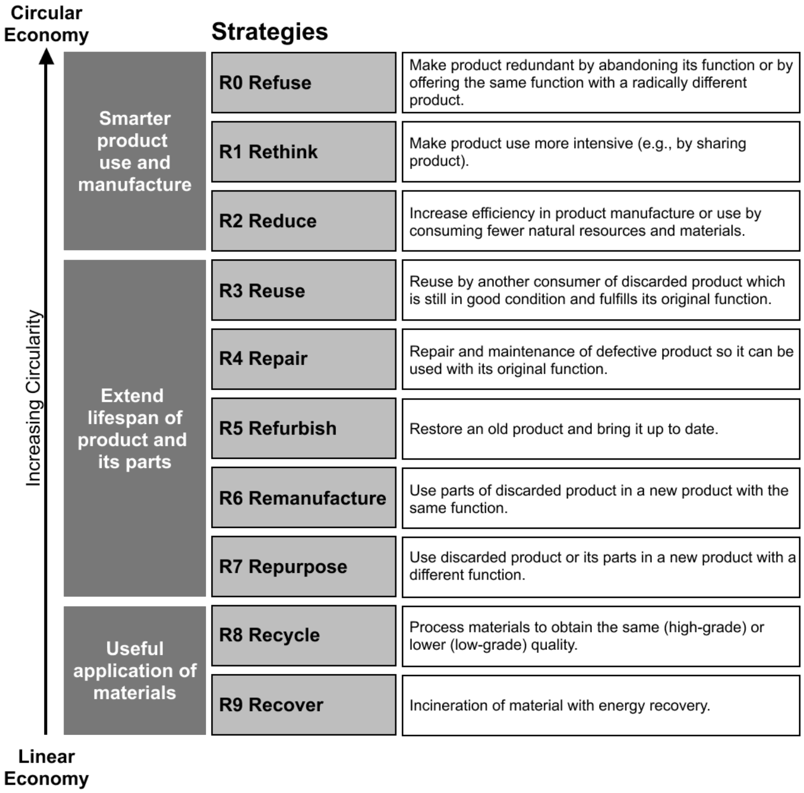

Circular Economy
Strategies
Increasing Circularity
Linear Economy
Smarter product use and manufacture
R0 Refuse
Make product redundant by abandoning its function or by offering the same function with a radically different product.
R1 Rethink
Make product use more intensive (e.g., by sharing product).
R2 Reduce
Increase efficiency in product manufacture or use by consuming fewer natural resources and materials.
Extend lifespan of product and its parts
R3 Reuse
Reuse by another consumer of discarded product which is still in good condition and fulfills its original function.
R4 Repair
Repair and maintenance of defective product so it can be used with its original function.
R5 Refurbish
Restore an old product and bring it up to date.
R6 Remanufacture
Use parts of discarded product in a new product with the same function.
R7 Repurpose
Use discarded product or its parts in a new product with a different function.
Useful application of materials
R8 Recycle
Process materials to obtain the same (high-grade) or lower (low-grade) quality.
R9 Recover
Incineration of material with energy recovery.

Figure 4.10: 9R strategies, adapted from Potting et al. (2017)

From a practical perspective, the emphasis is on three core principles: **efficiency**, which entails achieving maximum benefit with minimal effort; **consistency**, which involves adopting circular economy practices and alternative production methods; and **sufficiency**, which focuses on behavioral change to promote reduced consumption. Any software that addresses one or more of these aspects is considered to have an ecological purpose according to this methodology.

Practical Guide: 'Text Template for Sustainable Impact'

This methodology provides a clear, standardized template for defining the sustainable value proposition to articulate the purpose of the software in detail.

Template "Sustainable Value Proposition":

The sustainability value of the [Software Application] lies in its commitment to the [Sustainability Dimension] dimension. By [Functions and Features], the software contributes to achieve [Sustainable Benefits], resulting in [Sustainable Outcome]. This software technology supports companies in [Strategic Goal towards Sustainability].

4.4 Layer of Software Design

The proposed methodology focuses in the second building block on the tool, i.e. the value creation of software. The focus lies on the design, which contributes to re-engineering the software technology to ensure that it follows sustainability principles. However, this work does not contribute to the research stream of developing software that consumes, for example, less electricity. Instead, this thesis outlines an approach to ensuring that sustainability is integrated from both a technical quality and a technical innovation perspective. Thus, the primary focus is on integrating the three dimensions of sustainability into the software's quality attributes. The secondary focus examines whether the software's level of technical innovation supports sustainability.

4.4.1 Sustainable Quality Attributes

This chapter refers to Publication III.

To ensure that the integrated sustainability software performance framework (ISSPF) is seamlessly integrated into the herein proposed methodology, the key requirements that led to the development of the aforementioned model are presented in detail below in order to ensure a tangible understanding of the entire concept.

Requirement 1:

Sustainability is increasingly recognized as a critical dimension of software design, especially in the context of non-functional requirements (NFR). This thesis builds upon recent research, acknowledging that sustainability forms an essential part of software quality attributes (Venters et al., 2014). According to established studies, these attributes influence the longevity and impact of software in terms of its environmental, economic, and social implications. Sustainable design, therefore, is not only a desirable feature but also an indispensable aspect of high-quality software development. As Aljarallah and Lock (2019) noted that poor quality often results in substandard software, leading to lower adoption rates and consequently hindering sustainability goals. This relationship between quality and sustainability underscores the need for a well-defined framework that integrates sustainability into software design. Gerstlacher et al. (2022) extended this concept by examining whether existing quality models can serve as the foundation for green and sustainable software, which aligns with sustainability principles without sacrificing operational efficiency. This thesis advocates the perspective that sustainability is not merely an additional NFR contributing to software quality. Instead, sustainability can be identified within all existing NFRs, suggesting that NFRs are integral to software quality, which in turn is part of software sustainability. Based on this perspective, the proposed methodology applies the model of integrating software development layers into existing sustainability assessment frameworks (Schott and Ovtcharova, 2024).

This approach focuses on the so-called software sustainability attributes, aiming to ensure the long-term use and adaptability of digital systems under evolving environmental conditions, user requirements, and societal well-being (Moreira et al., 2023). In this context, sustainability is not just another quality attribute but an evolution of established quality characteristics. Venters et al. (2014) suggested that sustainability cannot be seen as a standalone or composite attribute; instead, it is a progression of existing quality elements that connect with the three pillars of sustainability: environmental, economic, and social.

<u>Requirement 2:</u>

ISO/IEC 25010, an international standard for software quality, plays a pivotal role in this discourse. This model evaluates software based on two key dimensions: product quality and quality in use. Although this thesis incorporates both dimensions, it prioritizes product quality attributes because they directly influence the broader outcome of software performance, including context coverage (functionality) and user satisfaction (usability). The decision to use ISO 25010:2023 stems from its comprehensive nature, surpassing previous models like ISO 9126, McCall's, Boehm's, and others, as shown in Chapter 2.1.1. Its detailed approach offers better support for software sustainability characteristics, which other researchers also emphasize (Adewumi et al., 2015; Aljarallah and Lock, 2019).

<u>Requirement 3:</u>

One critical modification in ISSPF is the separation of efficiency into two categories: performance efficiency and resource/energy efficiency. This distinction reflects the growing importance of energy consumption in sustainable software design. By adopting ISO/IEC 25010, the proposed methodology is based on nine quality attributes, as listed in Figure 4.11.

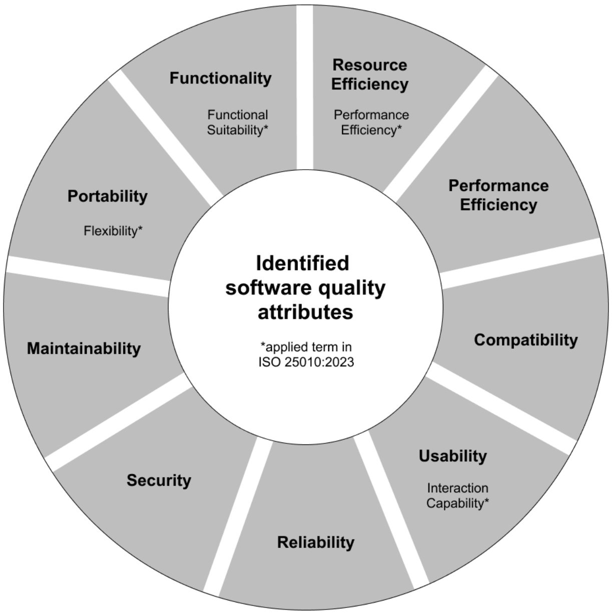


Figure 4.11: Software quality attributes for the proposed methodology

As Table 4.1 shows, the ISSPF consists of two dimensions and one type of value. One dimension reflects sustainability aspects, using the triple bottom line principle. The other dimension covers the development layers, differentiating between frontend, backend, and full-stack. The values of the ISSPF are covered by applying the attributes shown in Figure 4.11. This setting leads to a fourth requirement.

<u>Requirement 4:</u>

The interdependence of software quality attributes and sustainability is a key finding of this research. Moreira et al. (2023) asserted that these attributes are not isolated but rather interconnected. A sustainable quality attribute that primarily focuses on the frontend can also influence the backend development process. Similarly, ecological characteristics can have economic implications, reinforcing the need for a flexible and holistic approach to sustainable design. Until a unified understanding of sustainable software is established in academia and industry, various approaches will continue to pave the way toward more sustainable solutions.

Table 4.1: Basic structure of ISSPF, based on Schott and Ovtcharova (2024)

Dimension 1 (x-axis)	**Development Layers**	Frontend, Backend, Full stack
Dimension 2 (y-axis)	**Triple Bottom Line**	Social, Economical, Ecological
Values	**Quality Attributes**	Functionality, Resource Efficiency, Performance Efficiency, Compatibility, Usability, Reliability, Security, Maintainability, Portability

By positioning sustainability as an evolution of existing software quality attributes, this chapter outlines the significant role that sustainability plays in the broader context of software development. Not only does it ensure that software remains adaptable and durable, but it also emphasizes the reciprocal relationship between sustainability and quality—one that ultimately drives innovation and resilience in software systems.

In the subsequent sections, the specific attributes will be analyzed in detail, their relationship to sustainability refined, and categorized according to the three defined layers of development. Figure 4.12 presents an initial insight into the resulting categorization.

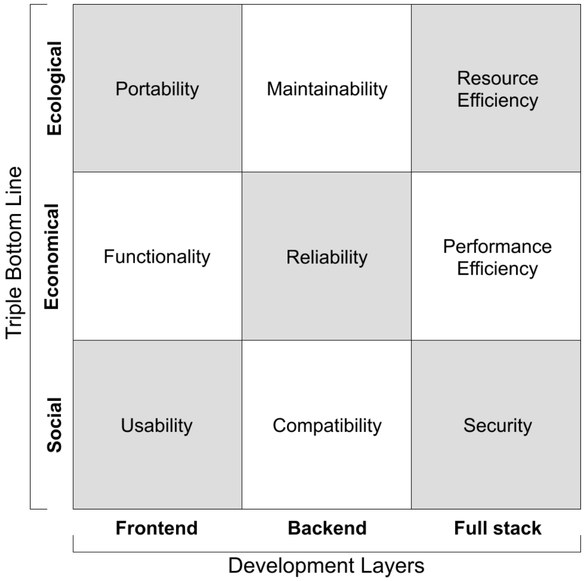


Figure 4.12: Categorization of quality attributes, based on Schott and Ovtcharova (2024)

Functionality

Functionality, also known as functional suitability, is a core attribute of software quality, defined as the degree to which a software product provides functions that meet stated and implied needs when used under specified conditions. Functional suitability encompasses three sub-attributes: functional appropriateness, correctness, and completeness. While functional appropriateness assesses whether the software provides the necessary features and functionalities relevant to performing the user tasks, functional correctness ensures that the functions of the software perform in a correct way and provide valid results. The third sub-attribute, functional completeness, ensures that all tasks the software is meant to be used for can be performed by the software, i.e., no function is missing. These sub-attributes collectively ensure that the software's functionality effectively fulfills its intended purpose

(Condori-Fernandez and Lago, 2018; International Organization for Standardization, 2023; Moreira et al., 2023).

The relevance of this attribute lies in its alignment with user requirements to achieve user satisfaction, loyalty, and operational efficiency. Therefore, functionality is demanded by stakeholders (Aljarallah and Lock, 2019). Software that meets functional requirements can streamline processes, enhance productivity, and open new business opportunities, contributing to an organization's overall success and growth (Khalifeh et al., 2023). This perspective causes the functionality of software to become an economic factor. Researchers have confirmed that functionality is a key factor in determining software acceptance andsales. It also serves as an indicator of the targeted customer segment and the potential size of the final market (Hui and Tam, 2002).

In the context of technical implementation, this model assigns functionality to the frontend of software. “Functionality itself can determine usability; if the functions provided do not match task requirements, a system will not be usable” (Goodwin, 1987). This interconnection causes functionality to be dependent on the user, who in turn interacts with the software via the frontend.

Reliability

Reliability is a critical attribute of software quality defined as the degree to which a system, product, or component performs specified functions under specified conditions for a specified period of time (International Organization for Standardization, 2023). Essentially, it ensures that the software behaves as expected, maintaining continuity of the correct service (Venters et al., 2014). Reliability can be divided into four sub-attributes. First, faultlessness indicates that the software operates without errors when used under the intended conditions. Second, availability refers to the readiness of the software to provide the correct service (Venters et al., 2014). Third, fault tolerance refers to a system's ability to continue functioning correctly even when some of its components fail, and fourth, recoverability is assessing how quickly and efficiently a system can recover from failures and return to normal conditions (Somani and Vaidya, 1997).

The assessment of reliability in relation to the triple bottom line principle of sustainability shows an increased affiliation with the economic aspect (Gerstlacher et al., 2022). From an economic perspective, reliable software is essential for reducing downtime and the associated costs of fixing errors or recovering from failures. Dependable software minimizes disruptions in business operations, leading to better financial performance. It reduces the need for frequent maintenance and support, lowering operational costs and enhancing overall productivity. By ensuring high availability and fault tolerance, businesses can avoid significant financial losses caused by system outages and failures (Khalifeh et al., 2023).

Reliability is a backend attribute in the context of the three development layers, as this layer is responsible for the system's continuous operation. Fault tolerance mechanisms and efficient recoverability are essential for maintaining the integrity and availability of backend services.

Performance Efficiency

Performance efficiency is a vital software quality attribute that evaluates a system's responsiveness to user interactions and its execution efficiency. It is characterized by the extent to which a software product executes its functions within specified timeframes and throughout limits while efficiently utilizing resources under defined conditions (International Organization for Standardization, 2023; Kocak et al., 2015). According to ISO 25010:2023, sub-attributes of performance efficiency include time behavior, resource utilization, and capacity. The first sub-attribute assesses the system's response time to user requests (Albertao et al., 2010), whereas the second evaluates how effectively the software leverages resources. The third sub-attribute, capacity, provides information on the maximum limits at which the software still meets the requirements (International Organization for Standardization, 2023). All these sub-attributes provide insights into the readiness of a software for use and are considered usage-related properties (Albertao et al., 2010). Due to their direct indication of the productivity level of a software, the attribute "Performance Efficiency" is understood as an economic attribute, offering essential insights into the operational efficiency and economic value of the software. The ecological component in this attribute has been extracted by dividing it into performance efficiency and resource efficiency and will be examined separately in the following.

In this thesis, performance efficiency is regarded as a full-stack attribute because it influences the entire system. By analyzing indicators like the average number of system users per time unit or the average page loading time, bottlenecks can be identified, causing optimization of the frontend and backend processes to enhance the overall performance of the software system (Ashanin, 2019).

Portability

Following the definition of ISO 25010:2023, portability is defined as the degree to which a software product can be adapted to changes in its requirements, usage context, or system environment (International Organization for Standardization, 2023). It is the ability of a system to operate across different computing environments without requiring extensive modifications. Based on this understanding, portability is often referred to as flexibility. The defined key sub-attributes include adaptability, scalability, installability, and replaceability. All these related attributes ensure a broader usability of the software, resulting in an extended lifecycle. By being adaptable to different computing environments, portable software enhances flexibility and reduces the need to develop multiple versions. This ensures that the software can meet diverse user needs and remain effective as the technology evolves (Albertao et al., 2010). As a result, portability contributes to reducing electronic waste and resource consumption, which classifies it in the first place as an ecological attribute (Albertao et al., 2010; Khalifeh et al., 2023). According to the development layer, the attribute of portability is primarily associated with the frontend layer. This attribute plays a critical role in scenarios in which software must be migrated or adapted to different user interfaces, ensuring a seamless transition across platforms. Portability encompasses the capacity of the frontend to adjust varying end-user devices and environments, such as migrating from iOS to Android. This flexibility ensures that end users can experience the same surface across multiple devices.

Maintainability

Maintainability is a fundamental attribute of software development and is essential for ensuring software sustainability. According to ISO 25010, maintainability is "the degree of effectiveness and efficiency with which a product or system can be modified to improve it, correct it, or adapt it to changes in the environment and in requirements" (International Organization for Standardization, 2023). Venters et al. (2014) further defined maintainability as the ability to undergo modifications, improvements, and corrective actions. This means that a system must support changes, whether due to evolving business needs or the correction of existing bugs, with the goal of easily updating features and technologies.

Five sub-attributes contribute to this software sustainability attribute, each playing a decisive role in terms of the maintainability of a software system. Modularity, reusability, analyzability, modifiability, and testability represent the sub-attributes of maintainability. They are not isolated but influence each other. A brief sub-attribute profile based on ISO 25010:2023 is given in Table 4.2.

Maintainability is classified as an ecological attribute in this thesis. Therefore, Khalifeh et al. (2023) stated that software that is easy to maintain and update can have a longer lifespan, reducing the need for creating new software and thereby lowering the environmental impact of software development.

Table 4.2: Sub-attributes of maintainability, adapted from International Organization for Standardization (2023)

Sub-Attribute	Definition
Modularity	… refers to the ability of software to be divided into individual parts that can be reassembled in various ways, creating new value and servicing for new use cases
Reusability	…often rated as its own software sustainability attribute due to its importance (Aljarallah and Lock, 2019), this sub-attribute focuses on the ability to be reusable itself or at least to re-use various components and functions of the developed technology in other software applications or use cases.
Analyzability	… describes the ease with which software can be analyzed, with a focus on identifying the causes of errors as precise and quickly as possible.
Modifiability	… expresses the ease with which a software system can accommodate changes, with minimal sacrifices.
Testability	… ensures that the software is easier to understand, more efficient in order to check for faults, and thus easier to update.

Maintainability is also seen as an attribute influencing the backend of the software design. A microservice architecture, for example, plays a crucial role in enhancing maintainability, while solid documentation is necessary. Furthermore, microservices are well known for their "superior scalability and flexibility compared to monolithic designs" (Soongpol et al., 2024). Majuntke and Obermaier (2023) provided further detailed information on the importance of microservice architecture in the context of reaching sustainable software.

Resource Efficiency

Alongside performance efficiency, this methodology also adds the attribute of resource efficiency to the collection of software sustainability attributes. This approach is not new but will be examined below due to its relatively low profile within this context. In 2015, Kocak divided the quality attribute of efficiency into performance, energy, and resource efficiency. Energy efficiency refers to how well the software performs regarding energy consumption under specified conditions, focusing on minimizing the energy used during operation, whereas, resource efficiency pertains to how effectively the software utilizes its resources when performing its functions and handling workloads, emphasizing optimal resource usage to ensure efficient performance (Kocak et al., 2015). Following another approach but with a similar conclusion, Gerstlacher et al. (2022) also announced that the performance efficiency attribute from the ISO framework needs to be enriched by a second attribute, namely greenability. This characteristic should cover energy and resource efficiency. Khalifeh et al. (2023) revealed that ecologically efficient software reduces the consumption of computer resources such as computing power, which can lead to a smaller ecological footprint. Consequently, resource efficiency, including energy efficiency, is seen as an attribute that influences the ecological performance of software technologies.

Concerning the technological layer, a full stack assignment is established here. Energy efficiency is predominantly associated with backend operations, focusing on optimizing energy consumption through efficient server management and processing. Conversely, resource efficiency is more relevant to frontend development. It aims to streamline user interactions by minimizing the number of clicks required to achieve a desired outcome.

Usability

The term usability can be differentiated into two approaches: a product-oriented view focusing on ease of use and a broader human factor approach, which defines usability as the ability to use a product for its intended purpose and within its intended environment.

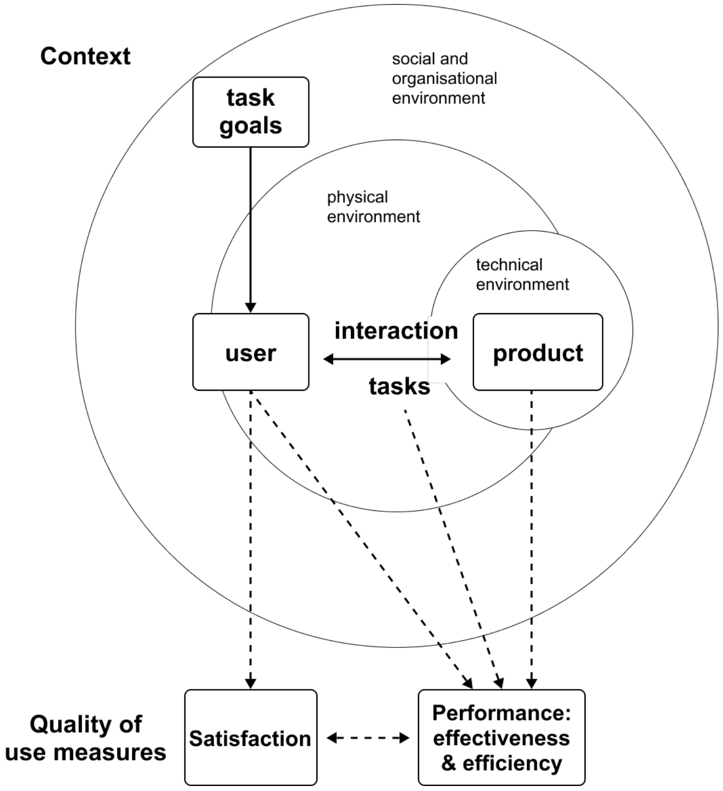


Figure 4.13: Usability based on the human factor approach (Bevan, 1995)

The product-oriented approach aligns with traditional software engineering, while the broader view emphasizes user-centered design. The illustration by Bevan (Figure 4.13) highlights the relationship between software product quality and its context. It shows that product quality is influenced by the technical environment in which it operates and the needs and expectations of stakeholders, including users and organizations. This interaction emphasizes the cruciality of designing software that aligns with technical requirements and stakeholder goals. In total, usability should generally be seen as a design attribute and a quality objective, ensuring the product's real-world effectiveness (Bevan, 1995).

In this chapter, the concept of usability as a quality attribute will be further explained, which is also referred to as interaction capability. Both terminologies describe the state of how intuitively and smoothly users can use software. The aim of interacting with the user interface is to obtain specific information or perform certain actions. Albertao et al- (2010) defined usability as a usage-related attribute that brings user-friendly features in the position of influencing a higher degree of usability. Furthermore, they mentioned that usability is the "system's ability to serve people regardless of location, experience, background, or the type of computer technology used" (Albertao et al., 2010).

In total, eight sub-attributes are highly associated with the degree of the software's usability, based on ISO 25010:

- Appropriateness recognizability
- Learnability
- Operability
- User error protection
- User engagement
- Inclusivity
- User assistance
- Self-descriptiveness

Since the degree of usability is evaluated in direct interaction and exchange with the user, this attribute can be assigned to the social dimension of sustainability. Software that follows a user-friendly and accessible approach markedly enhances customer satisfaction, user diversity, and overall social well-being (Khalifeh et al., 2023) while contributing to greater inclusion in digital environments (Albertao et al., 2010). The description "interacting with the user interface" underscores that usability is inherently tied to the design of the software's frontend. This connection is central to the field of human-computer interaction (HCI), which focuses on optimizing frontend aspects such as usability, accessibility, and user experience. By understanding user preferences, cognitive processes, and behaviors, HCI aims to create interfaces that seamlessly adapt to the needs of users (Paneru et al., 2024).

Compatibility

Compatibility refers to the ability of software to operate and communicate correctly across various environments, including different operating systems, devices, and other components. Compatibility is crucial for user satisfaction, as it allows the software to function in multiple contexts without requiring notable adjustments (International Organization for Standardization, 2023; Motta et al., 2019). The compatibility attribute can be broken down into the sub-attributes of coexistence and interoperability. Interoperability, a critical software system property, presents a significant technical challenge. Deploying applications across diverse fields and purposes requires support for varying components, each designed with features and functions tailored to specific application objectives. This ensures that the software can effectively communicate and operate within different systems, enabling seamless data exchange and integration (Motta et al., 2019).

Compatibility is crucial for social sustainability because it promotes inclusivity and accessibility by ensuring that software can operate across various systems and devices. This reduces the digital gap, allowing more people—regardless of their technical resources or abilities—to access and benefit from the product or service (Khalifeh et al., 2023). For a digital application to run on various environments and communicate seamlessly, a compatible backend is essential. This is why compatibility is mainly categorized as a backend issue.

Security

In recent years, the frequency of attacks on information systems has risen steadily, highlighting the growing importance of robust security measures. Security, as outlined by ISO, refers to the degree to which a product or system protects against cyberattacks and safeguards information, ensuring that access to data is appropriately restricted based on authorization levels. In software development, this is closely linked to safety, which involves preventing catastrophic consequences for users and the environment, emphasizing the importance of strong security practices (International Organization for Standardization, 2023; Venters et al., 2014).

The key sub-attributes of security include confidentiality, integrity, non-repudiation, accountability, authenticity, and resistance. Confidentiality ensures that sensitive information is accessible only to those authorized, while integrity maintains data accuracy and consistency. Non-repudiation prevents denial of actions performed by an entity, and accountability enables tracing actions to responsible parties. Authenticity verifies the identity of users and systems, and resistance refers to the system's ability to withstand attacks (Mohamed, 2020).

Following these descriptions, security markedly contributes to social sustainability by protecting user privacy, ensuring data confidentiality, and preventing unauthorized access. Multiple security services and tools need to be implemented in software systems to achieve a certain level of security. Figure 4.14 shows the selection. Accordingly, security measures are a core component of software design and development because they help build trust and maintain a company's reputation, contributing positively to society (Khalifeh et al., 2023).

In software architecture, security is addressed across the full stack. On the frontend, security measures include implementing authentication, authorization, and encryption of passwords and content. On the backend, security is reinforced through audits, penetration tests, secure connections, and data encryption, providing a comprehensive defense against potential threats.

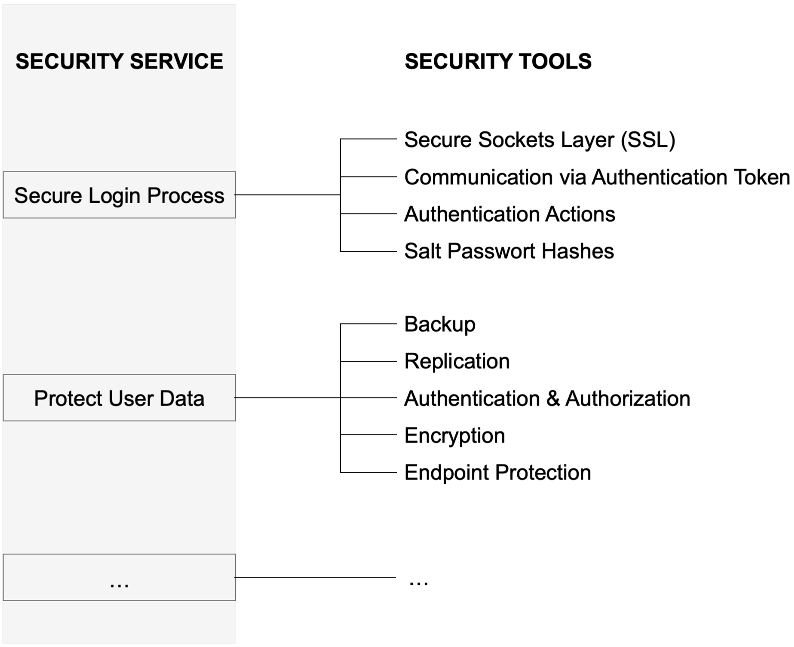


Figure 4.14: Examples of software security services and tools

Summary of the Integrated Sustainability Software Performance Framework

The ISSPF classifies nine software quality attributes according to the triple bottom line principle and the three development levels. This indicates that sustainability in software development is not seen as an extra dimension but can be recognized in all current quality attributes. This approach aims to ensure that software development aligns with comprehensive sustainability goals. Since the ISSPF was initially created as a stand-alone framework, it also features a scale for evaluation alongside categorization. This scale can be used to evaluate individual attributes in terms of their level of maturity or to highlight any missing or poorly implemented attributes during the review process. Therefore, reflection and evaluation should be conducted regularly throughout the entire software lifecycle.

This maturity assessment is not part of the methodology presented here, as the ISSPF is integrated into the holistic re-engineering methodology. Thus, a checklist is proposed for practical use. The purpose of this checklist is to systematically and thoroughly assess the existing software regarding quality and sustainability attributes.

Practical Guide: 'Checklist for Software Quality Attributes'

This checklist serves as a structured and practical tool for checking the implementation of software quality attributes. As Table 4.3 shows, each attribute is accompanied by a high-level question to ensure relevance and clarity. The summary of the checklist is designed to identify strengths, weaknesses, and areas for improvement, supporting informed decision making throughout the software lifecycle. By systematically addressing these quality attributes, the checklist promotes the development of sustainable, efficient, and user-centered software solutions.

Table 4.3: Checklist for software quality attributes

Nr.	Software quality attribute, incl. high-level questions				Checkmark appropriate column	
					Yes	N/A
1	Economical	FE	**Functionality**	Does the software meet all required functional specifications?		
2	Economical	FS	**Performance Efficiency**	Does the software perform efficiently under expected conditions?		
3	Economical	BE	**Reliability**	Does the software remain stable and recover effectively from failures?		
4	Social	BE	**Compatibility**	Is the software compatible with the required platforms and systems?		
5	Social	FE	**Usability**	Is the software intuitive and easy for users to operate?		
6	Social	FS	**Security**	Does the software have adequate measures to protect against security threats?		
7	Ecological	FS	**Resource Efficiency**	Does the software optimize resource and energy consumption effectively?		
8	Ecological	BE	**Maintainability**	Can the software be easily updated, modified, or maintained?		
9	Ecological	FE	**Portability**	Can the software be easily transferred to other environments or platforms?		

SUMMARY		**In total:**	**_ _ / 9**
Amount of implemented attributes by dimension:	Economical _ _ / 3	Social _ _ / 3	Ecological _ _ / 3
	Front End (FE) _ _ / 3	Back End (BE) _ _ / 3	Full Stack (FS) _ _ / 3

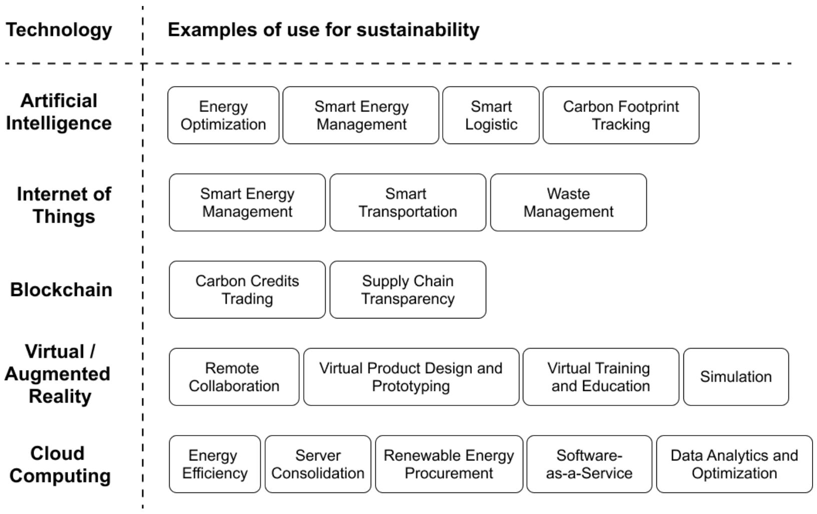
4.4.2 Innovation Attributes

The software innovation attribute segment represents the forward-thinking, creative, and outside-the-box aspect essential in re-engineering software technologies. Technology innovation is crucial for sustainable software applications because it enables continuous adaptation to evolving user needs, industry standards, and environmental challenges. The proposed methodology addresses innovation by integrating cutting-edge technologies and delivering exceptional user experience. This ensures that the software remains at the forefront of technological advancements, which are crucial for long-term sustainability. By leveraging the latest technologies and UI/UX design trends, the software not only meets current needs but also reduces the likelihood of becoming obsolete, thereby minimizing the need for frequent revisions or replacements. This proactive approach contributes to a more sustainable digital ecosystem by extending the software's lifespan and reducing resource consumption and waste over time. Unlike sustainable quality attributes, which remain constant, software innovation attributes evolve rapidly. Therefore, regularly updating these attributes using tools like the Gartner Hype Cycle to maintain the software's relevance and sustainability is crucial. Regular checks, ideally on an annual basis, ensure that the software continues to operate efficiently and sustainably in a constantly changing technological landscape.

Cutting-Edge Tech Stack

This section explores how well the implemented tech stack aligns with cutting-edge technological advancements. It evaluates how well the software application incorporates the latest technologies and offers a framework for analyzing its technological advancement. Cutting-edge technologies encompass the latest technologies, which are often transformative and revolutionize their respective environments even though they have not reached full maturity. These technologies, with transformative characteristics, are supposed to have a higher level of risk due to their lack of experience and data. Still, they have enormous potential to elevate competitive advantages, enhance software performance, and provide an improved foundation for scalability and security. Such benefits can manifest by creating new BMs, processes, functionalities, or features. Examples of cutting-edge technologies are announced every year in the well-known Gartner-Hyper Cycle. In 2024, nominated technologies include advancements in platform development, AI-supported development, intelligent applications, cloud platforms, sustainable technologies, and the democratization of generative AI (McCartney, 2023). Although some cutting-edge technologies operate across industries, they can also differ between industries due to their scope of impact. In mechanical engineering, current trends include software related to additive manufacturing, robotics software, and applications for digital twins, as well as virtual and augmented realities. Additionally, AI and IoT play a significant role (Horner, 2024). The continuous adoption, development, and integration of these emerging technologies enables organizations to remain at the forefront of their industry, creating new value through enhanced functionality and thus creating a positive framework for ensuring the long-term and high-frequency use of the software. This approach drives innovation and contributes to the sustainable evolution of software solutions.

Table 4.4: Potential of cutting-edge technologies for sustainability, adapted from Abdelkafi et al. (2023)



Technology	Examples of use for sustainability				
Artificial Intelligence	Energy Optimization	Smart Energy Management	Smart Logistic	Carbon Footprint Tracking	
Internet of Things	Smart Energy Management	Smart Transportation	Waste Management		
Blockchain	Carbon Credits Trading	Supply Chain Transparency			
Virtual / Augmented Reality	Remote Collaboration	Virtual Product Design and Prototyping	Virtual Training and Education	Simulation	
Cloud Computing	Energy Efficiency	Server Consolidation	Renewable Energy Procurement	Software-as-a-Service	Data Analytics and Optimization

To make the impact of cutting-edge technologies on innovation and sustainability more tangible, Table 4.4, extracted from Publication I, provides some ideas on how technologies can be used to create positive, sustainable outcomes. The table includes AI, IoT, blockchain, VR, AR, and cloud computing. AI and extended realities (XR) are chosen to showcase their potential for sustainability in greater detail, while the latter partly refers to Publication II.

In 1956, the term "artificial intelligence" was coined at the Dartmouth Conference by pioneers like John McCarthy, Marvin Minsky, Allen Newell, and Herbert A. Simon. Although numerous definitions of AI exist in research and practice, many perspectives refer to it as "the capacity of an information-processing system to adapt to its environment while operating with insufficient knowledge and resources" (Wang, 2019). Therefore, AI enhances the innovative aspects of software by automating repetitive tasks, extracting insights from vast and complex data, and integrating large-scale computational resources. These capabilities enable AI-driven software to tackle complex problems and create innovative solutions more efficiently. Moreover, AI can detect patterns and trends from unstructured data (e.g., images, text), facilitating more innovative features and functionalities that surpass human intuition or traditional computational methods (Nishant et al., 2020). AI, as the gateway toward more intelligent applications (Soongpol et al., 2024), allows software not only perform complex operations more efficiently but also to evolve and self-improve over time. These features foster innovation by enabling new functionalities, optimizing processes, and creating adaptive systems capable of addressing real-time challenges and opportunities (Wang, 2019). This wide range of opportunities serves as a critical tool in mitigating global GHG emissions, delivering environmental benefits that far outweigh their minimal contribution to those emissions. AI enhances sustainability by optimizing resource use, improving the efficiency of industrial operations, and reducing energy waste (Abdelkafi et al., 2023). Furthermore, AI improves decision making by eliminating human biases, mitigating environmental risks, and fostering better environmental policies and governance (Nishant et al., 2020).

Continuing with the second example, XR, an umbrella term for VR and AR, enables humans to combine the physically familiar world with a simulated, data-based digital world. Slater describes it as a technology that empowers the end user with a high quality and quantity of sensory information (Slater, 2009; Suh and Prophet, 2018, p. 2), while mapping physical, spatial, and visual dimensions in this computer-generated simulation of reality (Handa et al., 2012). XR enables people to see and experience previously invisible and tangible things (Ovtcharova, 2022). Due to its technological features, including immersion, interaction, imagination, and intelligence (Burdea and Coiffet, 2003; Häfner, 2021), XR technologies greatly contribute to sustainability by promoting environmental awareness and facilitating pro-environmental decision making through immersive, multi-sensory experiences. These experiences enable optimized value delivery, ensuring that users understand the importance of sustainability in a compelling way. XR's global accessibility enhances social equity by offering inclusive opportunities across geographical boundaries. Virtual prototyping within XR reduces the need for physical resources and contributes to lower material consumption. In addition, virtual meetings and services minimize the need for travel, thereby reducing carbon emissions.

XR also supports social sustainability by offering virtual training programs that are accessible to all, fostering skill development and social inclusion (Krodel et al., 2024). Finally, XR allows for precise communication, leading to leaner and more efficient processes that reduce resource waste. The potential impact of all these possibilities is currently still limited by costly hardware, the necessity of high-level specialists and the lack of standardized data exchange (Davila Delgado et al., 2020).

User Experience Excellence

The chosen terminology of “user experience excellence” in this context of software re-engineering refers to the strategic process of enhancing a software application to provide the highest level of a seamless, intuitive, and latest user experience. The objective of this dimension is to provide the best software interface to users, ensuring that the application remains relevant and valuable over time, is used frequently, and thus impacts the overall success of a digital application. On the one hand, cutting-edge technologies, as described in the previous section, often influence user experience design, as different technologies can provide different opportunities for interaction—for example, VR provides an immersive experience to the end user of the software. On the other hand, the advent of advanced technologies demands effortless user interactions and the cultivation of engaging experiences (Paneru et al., 2024). Nevertheless, several more aspects enrich the level of UX with excitement attributes. As of now, personalization and customization, game mechanics, and micro-interactions are quite popular within UX design. Nowadays, multiple options exist for individualizing the UX of software technologies. A distinction is mainly made between system-initiated personalization and user-initiated customization (Sundar and Marathe, 2010). The first option focuses on personalizing content that addresses the user's needs or searches. When content is tailored to user behavior or browsing history, users are likelier to remain engaged with the application, leading to an increased likelihood of return visits due to their satisfied experience. Furthermore, individualization can be applied not only to content but also to the look and feel of the digital application through customization. This aspect includes the possibility of applying corporate identity schemes to provide the user with different themes (e.g., background colors, fonts, and icon style) to select from to align them with personal preferences. In addition, the importance of ethical design as an element of UX excellence has emerged, focusing on prioritizing user trust and transparency. Furthermore, context-aware interaction design, which leverages sensors and location-based information, contributes markedly to creating more dynamic and relevant user interactions (Paneru et al., 2024). In total, customization is an excellent way to empower users to shape and enrich their experience with the digital application.

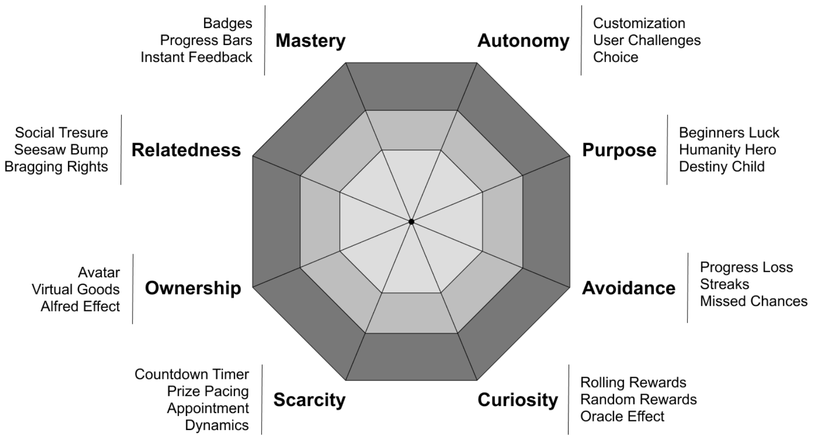


Figure 4.15: Gamification octalysis, adapted from Chou (2019)

Another instrument to elevate UX excellence is the inclusion of game mechanics, commonly referred to as ‘gamification’. Gamification is defined as "the intentional use of game elements for a gameful experience in non-game tasks and contexts" (Seaborn and Fels, 2015). Game elements, in this context, include patterns, objects, principles, models, and methods derived directly from game design. These elements are implemented through specific game mechanics, such as clear goals, levels, points, achievements or badges, leaderboards, and rewards. Figure 4.15 illustrates gamification octalysis, which includes the eight core human drivers and offers examples of game mechanics for use in engaging specific drivers. By incorporating these mechanics, UX design can foster greater user engagement, intrinsic motivation, and satisfaction, ultimately enhancing overall UX (Seaborn and Fels, 2015; Sharma et al., 2024). Gamification can enhance user experience, thereby contributing to the sustainable utilization of software while directly influencing sustainability by leveraging game-based mechanisms to raise awareness of and promote engagement with sustainable practices (Lee et al., 2013).

Micro-interactions can be added as a third example of achieving UX excellence. Micro-interactions play a vital role in improving UX by delivering instant feedback, guiding users through tasks, and making digital interactions feel more natural and engaging. An example of a micro-interaction is the "like" animation on social media platforms, where a heart icon briefly enlarges and changes color when tapped. This visual feedback not only confirms the action but also adds a layer of engagement. Such interactions reduce errors, guide users seamlessly, and contribute to a cohesive and enjoyable digital experience (Soegaard, 2024).

Practical Guide: 'KANO Model for Innovation Attributes'

Innovation attributes are herein defined as attributes that increase end user satisfaction based on the assumption that higher customer satisfaction leads to more sustainable software technology, especially regarding longevity and efficiency. The Kano model, developed in the 1980s by Noriaki Kano, is a tool frequently used for evaluation and reflection in research and practice, to assess software technology mainly from a business perspective. Therefore, within this methodology, the Kano model is suggested as a framework to guide and reflect the initial design of software functions. As shown in Figure 4.16, the Kano model describes functionality in its x-axis and satisfaction in its y-axis. In addition, the three main lines in the diagram. These indicate the categorization of functions into basic, performance, and excitement. While innovations can occur at all three levels, focusing on performance and excitement functions is particularly recommended, as these greatly impact user satisfaction along the y-axis.

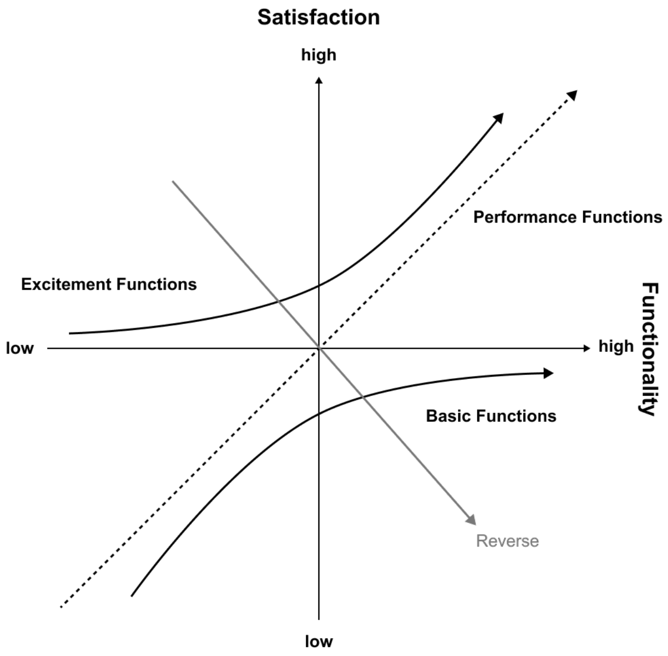


Figure 4.16: Kano model, adapted from Mayer (2012)

To make it more tangible, the following section offers a brief example of both technology segments of the methodology.

In the context of cutting-edge technology, basic functions include features like cloud integration, which are expected by default and do not increase satisfaction if fulfilled but cause dissatisfaction if absent. Performance, such as AI-driven personalization, directly contributes to satisfaction in proportion to its quality and effectiveness. Excitement requires elements such as XR integration or generative AI features to exceed user expectations, fostering delight and setting the product apart in the market.

For user experience excellence, the basic encompasses essential functionalities such as login mechanisms or responsive user interfaces, which form the foundation of usability. Performance functions include features such as content personalization, which, when well-executed, significantly enhance user engagement. Finally, excitement functions, such as gamification elements, introduce unexpected and innovative interactions that exceed user expectations, driving satisfaction and loyalty.

In summary, the purpose of applying the Kano model is to check whether software functions fulfill customer satisfaction and to ensure that the software includes both basic features and excitement attributes.

4.5 Layer of Software Enablers

This chapter discusses the role of organizational strategy (Section 4.5.1), resource management (Section 4.5.2), and market engagement (Section 4.5.3) as enabling dimensions for successfully implementing software technology in sustainable BMs as part of the re-engineering methodology. Therefore, software enablers are crucial execution factors that turn concepts into reality, drive innovation, and ensure seamless integration to achieve sustainable results.

4.5.1 Organizational Strategy

This chapter examines organizational strategy as a vital component of software enablers necessary for integrating software technology into sustainable BMs. Strategic re-engineering of software technology requires a comprehensive assessment of how well the software's purpose aligns with the organizational strategy. This alignment is crucial because BM success is inherently tied to the support it receives at the decision-making levels of an organization. However, effective management alone cannot realize lasting changes. It is imperative to:

- clearly communicate a coherent **corporate vision**
- empower **leadership** to drive transformation
- cultivate a **corporate culture** that supports innovation and sustainability.

By doing so, organizations can ensure that their software technology initiatives are effectively integrated and contribute to long-term business success.

Corporate Vision

The corporate vision acts as a core element of the organizational strategy. Its main task is to guide an organization by describing a future concept (El-Namaki, 1992). Through this direction, the company ensures that everyone involved has the same goal in mind. This common orientation creates a sense of belonging and identity within the company and teams. Accordingly, decisions are made, from management to staff level and from strategic to operational issues, toward the same understanding. Every project initiated should align with the corporate vision to contribute to the common goal, add value, and ensure that resources are used reasonably without running the risk of missing opportunities.

For the strategic re-engineering of software technology to be applied in the value proposition of a sustainable BM, practitioners must ensure that digitalization and sustainability are incorporated into the corporate vision. This requirement is necessary to run a successful and sustainable software application.

An example of such a vision for a start-up company in the financial sector could be as follows:

"Our vision is to revolutionize the finance industry by leveraging cutting-edge digital technologies to create accessible, transparent, and sustainable financial solutions. We aim to empower individuals and businesses to achieve financial well-being while fostering a greener, more equitable future."

This vision statement highlights the company's commitment to digital innovation and sustainability, setting a clear direction for growth and impact in the finance sector. Notably, although the vision exists first, it can be changed within the scope of responding to environmental changes (El-Namaki, 1992). The importance of a consistent and compelling vision in a company is also relevant because it prevents silo thinking and allows companies to drive a cross-departmental, targeted approach. When applied to software projects, companies must ensure that all new software development initiatives are clearly aligned with the corporate vision and strategic goals. This requires effective communication, collaboration across departments, and a robust governance framework to assess and guide software initiatives. A clear vision leads to well-defined objectives and clarifies the tasks required. As a result, the software developed is aligned with these objectives, reducing the likelihood of frequent revisions.

Leadership

The concept of leadership within an organization embodies encouraging, empowering, and appreciating behavior toward the team's efforts. As role model, leaders are responsible for embedding change as a positive element within the teams and company culture, thereby enhancing organizational agility (Kolasani, 2023). Change, as a key element of long-term success, should actively be integrated into daily business practices and core values. Change is also a key driver for initiating sustainable transformation and DT. This transformation can only be implemented at all levels when leadership fosters an innovative and diverse corporate culture, enhancing creativity and adaptability (Chen et al., 2024; Liu, 2022). Through these actions, leaders can cultivate an environment characterized by a sense of belonging and commitment, thereby fostering collective progress toward a shared objective. Leaders should also delineate a comprehensive roadmap, allocate tasks, and monitor progress against established milestones, all while ensuring transparent and open communication. This underscores that leadership operates not only on a strategic level but also at an operational level. Moreover, specific competencies are required for this role. These include strategic thinking, resilience, analytical acumen, a visionary and entrepreneurial mindset, emotional intelligence, and a sense of opportunities.

Beyond these general descriptions of leadership, its pivotal role in modern enterprises emphasizes additional key characteristics, particularly in guiding the integration of digitalization and sustainability. Chen et al. (2024) emphasized the importance of digital leadership in enhancing market-driven BM innovation and sustainable performance and advocated for developing digital competencies to effectively manage organizational transformations. Petrov et al. (2023) further underscored the necessity for leaders to combine digital skills with a sustainability mindset, ensuring long-term resilience by embedding ethical and environmental considerations into business strategies. Both studies confirm that the successful integration of sustainability and digitalization is critical for the strategic direction of organizations, requiring leaders to balance innovation with responsible business practices to meet the expectations of all stakeholders. Therefore, within this methodology, leadership takes over the responsibility that software technology is successfully implemented on an operational level. This encompasses critical decisions related to strategic goals, human and financial resources, and the technical orientation of the application.

Overall, the six key competencies a leaders should have to drive digital and sustainable transformation are listed in Table 4.5.

Table 4.5: Core leadership competencies (Schiuma et al., 2024)

Nr.	**Competence**	**Meaning**
1	**Visionary Thinking**	Setting a clear, future-oriented vision
2	**Strategic Management**	Aligning digital initiatives with business goals
3	**Innovative Mindset**	Encouraging creativity and adaptability

4	**Empathy and Collaboration**	Fostering a cooperative and supportive environment
5	**Resilience**	Maintaining stability during transformation
6	**Ethical Leadership**	Ensuring integrity and responsibility in digital entrepreneurship

Corporate Culture

Organizational strategy, as an enabler of software technology, must account for corporate culture. In 1985, Edgar Schein defined corporate culture as the underlying assumptions and beliefs shared among members of an organization, operating unconsciously and shaping the organization's fundamental perception of itself and its environment (Assoratgoon and Kantabutra, 2023). Corporate culture comprises the norms, values, beliefs, and perspectives that are practiced within the company and that influence the behavior of its employees. Cultural consensus largely determines whether a company´s employees support the development and application of software technology or whether active or unconscious aversion prevails within the team. In other words, the culture anchored in the organization has a direct impact on project implementation (Echterhoff, 2018; Leong et al., 2023). The risks associated with a mismatch during software re-engineering include employee dissatisfaction, increased resistance to change and adoption, and a decline in work ethic. Therefore, fostering a culture of diversity and innovation is crucial, as is creating an environment that promotes continuous learning throughout the software re-engineering process.

Incorporating sustainability and digitalization into corporate culture at an early stage is beneficial when considering the proposed value proposition for a sustainable BM. This can be supported by maintaining open communication, offering continuing education programs, and encouraging an open exchange between teams and hierarchical levels. Isensee et al. (2020) emphasized the reciprocal influence of culture and behavior and highlighted the importance of a cultural fit toward running digital and sustainable projects.

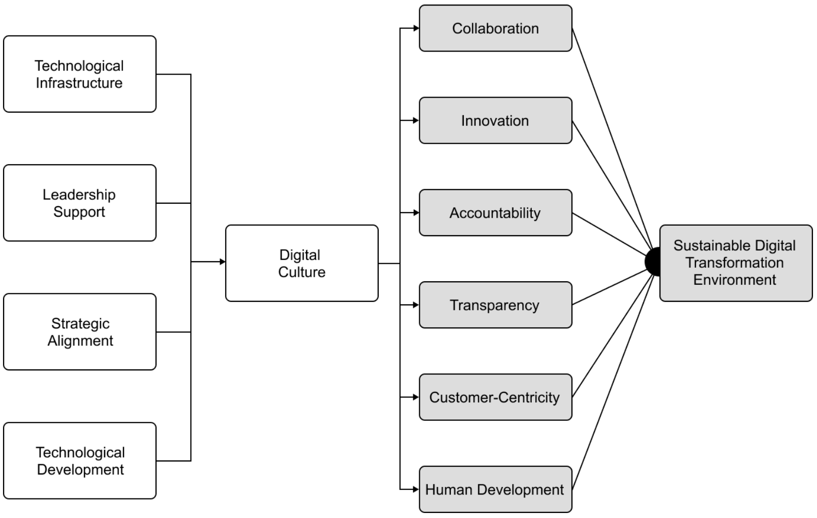


Figure 4.17: Role of digital culture in creating a sustainable digital transformation (Abdallah et al., 2022)

Figure 4.17 outlines the role of a digital culture in creating a sustainable DT environment, which can be used as an analogy for an environment conducive to software re-engineering linked to a sustainable BM. Following this, the foundation elements that contribute to creating such a required culture as the first step are input factors, like technological infrastructure (cf. Section 4.5.2), leadership support (cf. Section 4.5.1), strategic alignment (cf. Section 4.5.1) and technological development (cf. Section 4.4). Combining all these factors contributes to achieving a digital culture within the organization, following collective values and supporting change toward sustainability and digitalization. Tools in this context can only thrive if the underlying culture has a strong digital core. Further outcomes of a digital culture are, for example, efficient collaboration within and between teams, a high level of accountability as more team members are willing to take ownership, and customer- or user-centricity, ensuring that digital initiatives deliver real value (Abdallah et al., 2022).

Practical Guide: 'Prompt Questions on Organizational Strategy'

To ensure that the three dimensions—corporate vision, leadership, and corporate culture—fulfill their role as software enablers under the umbrella term of organizational strategy and within this methodology, Table 4.6 lists examples of questions intended to stimulate global thinking when conducting the methodology.

Table 4.6: Prompt questions on organizational strategy

Dimension	Relevance	Prompt Questions
Corporate Vision	To successfully implementing a software technology, it has to add value to achieve the corporate vision.	• Does the company communicate a clear vision, including digitalization and sustainability? • How does this software project support the overall corporate vision and strategic objectives? • How will the software technology enhance the market position?
Leadership	A strong leadership is required a) to develop the software technology effectively and b) to cultivate it within the team.	• Does the leadership encourage and promote change management? • Is the leadership reliable in its decisions? • Does the leadership have the required competences and characteristics? • How has the leadership communicated the importance and relevance of this project to the organization?

		• What resources and support have leadership provided to ensure the project's success? • Does leadership support the development and/or use of software technology?
Corporate Culture	Software technologies are more likely to succeed in open cultures that embraces change and agility.	• How does the software project align with the company's culture? • Do employees feel empowered to contribute ideas and solutions to digital and sustainable projects? • What cultural barriers exist that might hinder the success of digital and sustainable initiatives?

4.5.2 Resource Management

Software engineering, and likewise software re-engineering, is a human- and knowledge-intensive discipline (Wohlin et al., 2015). This indicates that software technology is heavily reliant on people, even though this reliance may diminish in the coming years due to the rising potential of generative AI. However, collaboration among teams and the expertise of each individual will remain essential for achieving successfully re-engineered software. Many of the related aspects are addressed in this methodology under the generic term 'Resource Management'. Thus, resource management is a crucial component of software enablers, providing valuable insights into the complexity of software technology and the level of expertise available within a company.

For the proposed methodology, and within this chapter, resource management emphasizes the following aspects:

- resource capacity,
- resource capability, and
- resource training and education.

Resource Capacity

The first element in the resource management segment focuses on ensuring that sufficient resources are available to implement a digital and sustainable application and, consequently, to facilitate the strategic reengineering process. Different types of resources should be considered within this methodology. The main focus is on human and financial resources, while technical resources, including hardware and software, as well as time and emotional conditions, should also be considered.

In this methodology, human resources pertain to the roles introduced in Section 4.2.1. The consideration goes beyond whether current employees can fill all the listed positions to encompass whether enough employees are available to manage the expected workload. This concern is influenced by several variables, such as the project timeline (e.g., are there enough resources to meet the delivery deadline?), the existence of competing projects running in parallel (e.g., does competencies have to work on multiple projects at the same time?) and the overall scope and complexity of the project. If internal staffing is inadequate, the company may need to initiate the recruitment of new team members at an early stage. However, in cases of employee shortage and significant time constraint—where additional resources are needed immediately beyond what can be managed through the recruitment process—the organization may need to engage external companies and incorporate them into the process, assuming this option remains previously unconsidered. In the case of a negative balance of human resources, compensation, either through recruiting or external service providers, has a strong impact on the financial resources of a project. The budget for running a software project must be planned in advance. This planning should account not only for the costs associated with developing the initial prototype but also for additional iterations, change requests, and further enhancements. To ensure sufficient financial resources are allocated, comparing the project with similar, successfully completed projects is advisable. Once the budget plan is established, responsibilities for specific budget shares should be clearly assigned, and expenditures must be continuously monitored.

In addition, further resource types listed at the beginning influence the budget. Specifically, technical resources—encompassing both hardware and software—are critical to the successful development and implementation of a digital project. This involves evaluating whether the existing technical equipment fulfills the scope of the software project. In terms of hardware, this means that the available workstations and servers are sufficient. Regarding software ensuring the availability of sufficient software licenses, for example, is necessary (e.g., task management tools).

Proper resource allocation and management are key to minimizing risks and ensuring the smooth progression of the project from initiation to completion. Moreover, preparedness for potential upcoming risks during the project is crucial. This means that companies need to consider risk management in the event of unexpected turnovers. This should include alternative human resources and the allocation of a buffer budget. By anticipating these risks and planning in advance, the project can avoid disruptions and increase the likelihood of a successful launch.

Resource Capability

Besides the evaluation, if enough resources are available to realize the software project successfully, the methodology also addresses whether the right resources are available. The aspect of “right resources” can be discussed from different angles. Two key perspectives are shared below to create a balanced understanding of the dimensions involved in resource capability.

On the one hand, ‘having the right resources’ means qualified specialists with the necessary knowledge, skills, and expertise are available to implement the transformation project. This concept is often referred to as human capital (Wohlin et al., 2015). In this regard, the focus should not only be on the theoretical skill set, listed in Figure 4.18 as domain expertise, but also on the practical experience already applied and proven in similar projects. This approach favors technical proficiency while assessing the readiness of human resources to master challenges and changes during the project’s lifetime. This includes, for example, the ability to manage potential risks and to effectively engage with different stakeholders to ensure the project´s overall success.

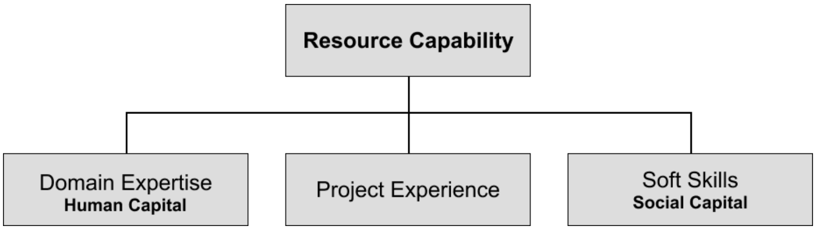


Figure 4.18: Resource capacity on its three components

On the other hand, having ‘the right resources’ also means having specialists with a common understanding. Human resources should, therefore, be committed to the company, team, and project, agree with the company's values, think in a solution-orientated way, and be willing to take responsibility for their work. This view is often called social capital (Wohlin et al., 2015). Further exemplary soft skill factors that influence the assessment of the capability of resources are adaptability, empathy, resilience, and willingness to learn.

The resource capability for the re-engineering process can be ensured only if the three components—domain expertise, project experience, and soft skills—align with the company's needs and culture.

Resource Education and Training

The third dimension of resource management extends the second dimension, resource capability. It highlights the readiness and promotion of education and training opportunities for various stakeholders to improve the teams´ abilities. The primary target group comprises all roles that directly impact the software project's success, usually the roles of business strategists, (technical) project managers, developers, and project owners. Organizations should engage in digital and sustainable transformation initiatives, particularly those utilizing software technologies and prioritize the continuous development of their human resources. Given the dynamic nature of technological advancements and evolving contextual conditions, the need for and support of ongoing professional development is a critical component of effective resource management. Particularly in the DT realm, staying up-to-date with technical advancements and development practices is crucial to enhancing skills in managing agile projects and stakeholder expectations. In addition, resource education and training are essential for the effective use of software technologies within the sustainability transformation context. This need is underscored by the ongoing development of sustainable business practices, which encompass ESG frameworks, circular economy principles, and sustainable innovation.

Consequently, the active promotion of relevant education and training programs has a positive impact on project execution and outcomes while also demonstrating the long-term benefits of human resources, highlighting improved innovation capacity, increased employee satisfaction, and loyalty. This resulting culture of continuous learning enhances the company’s reputation, drives employer branding, and contributes to long-term organizational growth.

Practical Guide: ‘Prompt Questions on Resource Management’

To ensure that the three dimensions—resource capacity, resource capability, and resource education and training—effectively serve as software enablers within the broader perspective of resource management in this methodology, Table 4.7 provides core questions designed to promote comprehensive consideration during the methodology's application.

Table 4.7: Prompt questions on resource management

Dimension	Relevance	Prompt Questions
Resource Capacity	To develop and run individual software solutions, enough resources are mandatory.	• Are there enough resources? • Are human resources available for realization? • Is there a budget allocated for upgrading infrastructure if needed? • Can existing resources be utilized? • Are there any potential resource constraints that could impact project delivery?
Resource Capability	To develop and run individual software solutions, the right resources are needed.	• Is the existing staff qualified for implementation and/or utilization? • What is the team's level of expertise in the specific domain relevant to the project's objectives? • Does the team show appropriate experience to master the project effectively? • Is the core team aligned with the “big picture” and do they share the corporate values?

		• To what extent has the team worked together on past projects, and what is the level of collaboration within the team? • Does the team demonstrate sufficient soft skills and therefore possess a strong foundation of social capital?
Resource Education & Training	For developing and further maintaining software technologies, the responsible resources must have access to knowledge and information.	• What training and development programs are in place to ensure team members are equipped to work on this project effectively? • Are concepts such as ‘change management’ actively and regularly communicated to the groups of people involved? • Are individualized and various types of knowledge transfer provided in the company? • Is a culture of constructive criticism actively practiced and is experience shared within the project team?

4.5.3 Market Engagement

For the proposed methodology, and as the third pillar within software enablers, the dimension of market engagement is defined as the proactive and strategic interaction with the end user of software technology. It includes end user support, continuous feedback loops, and marketing activities. The term “market” refers to the environment surrounding the end user. The environment in which the transformation process occurs may be either external or internal to the organization. When targeting an external market, the software technology is utilized by the customer, whereas for an internal market, it is employed within the organization, supporting its processes and employees. Furthermore, market engagement strategies must be precisely tailored to the type of end user and the specific characteristics of each environment to ensure maximum relevance and effectiveness.

End-User Support

Regarding both end-user types, the purpose of proactive end-user support is to demonstrate the functions and features of the implemented software technology and to highlight the benefits and associated improvements. Providing a detailed overview of the software functionality ensures that the user is well informed, trained, and equipped to utilize the software technology effectively and within the intended scope. This overview can be given by clear instructions, tutorials, or step-by-step guidelines. Conducting such sessions is important during onboarding and should be repeated and updated regularly to ensure that the end user stays informed about software updates and best practices. To maximize the effectiveness and address end user-specific scenarios, tailored and multi-channel support services could be used to provide precious end-user support. In fact, the degree of end-user support for utilizing the software directly correlates with their willingness to adopt the technology, which in turn enhances overall user satisfaction and user confidence.

Feedback-Loops

Another important characteristic of value-adding market engagement involves maintaining continuous feedback loops. The involvement of end users and the collection of inputs and insights are relevant throughout the software's lifecycle because they ensures that the software application remains responsive to evolving user needs. A high-level process flow is shown in Figure 4.19. In the first step, feedback needs to be collected from different stakeholders. This collection of feedback should be addressed in an iterative manner to keep the software technology, its features and functions aligned with the user's expectations. Various tools exist for running effective feedback loops, such as surveys, interviews, and real-time analytics. Ideally, the software application is designed such that the end user either generates feedback data naturally during regular use or is encouraged to provide feedback through methods that highlight the direct benefits to them, optimizing engagement and response quality (Hagiu and Wright, 2023). Additionally, feedback data should include both user-specific and product-specific data. This enables the development team to comprehensively understand the current maturity level and potential correlations.

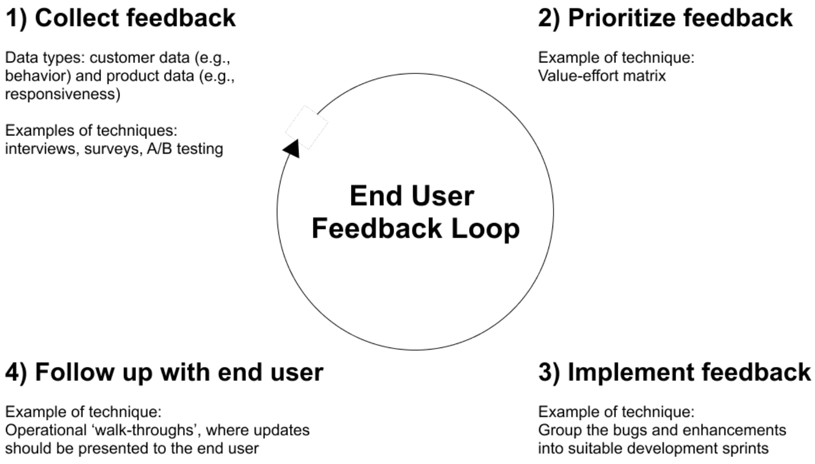

1) Collect feedback
Data types: customer data (e.g., behavior) and product data (e.g., responsiveness)
Examples of techniques: interviews, surveys, A/B testing
2) Prioritize feedback
Example of technique: Value-effort matrix
End User Feedback Loop
4) Follow up with end user
Example of technique: Operational 'walk-throughs', where updates should be presented to the end user
3) Implement feedback
Example of technique: Group the bugs and enhancements into suitable development sprints

Figure 4.19: End-user feedback loop

Following the cycle, in the next step, the project manager must analyze all gathered feedback points, including bugs (= errors in the current version) and enhancements (= ideas for improvements or extensions). Bugs should be resolved promptly as they impact software functionality, and enhancements must be prioritized according to their impact. If unclarity arises, it is advisable to contact the feedback submitter, if possible, before prioritizing to avoid any misinterpretation. To effectively prioritize these feedback points, a classic value–effort matrix, as depicted in Figure 4.20, is recommended, categorizing each point by its added value and the effort required. Continuing, the decisive factor for successful feedback loops as part of proactive market engagement does not lie in collecting feedback but in implementing suitable actions based on the collected feedback. Finally, after executing significant changes, the end user or the relevant stakeholders should be reconsulted to ensure that the change requests have been properly integrated into the software.

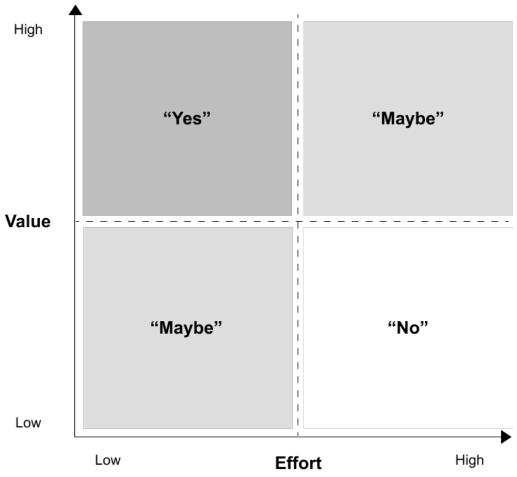


Figure 4.20: Schematic of the value–effort matrix

Marketing Activities

Besides user support and feedback loops, marketing activities represent a success-critical enabling function. Targeted activities should be designed to reach end users and activate their interest, while activities to establish internal advocacy should also be considered. In the domain of digitalization, exemplified here by the implementation of software technologies, and within the context of sustainability, marketing initiatives are inherently linked to educational efforts aimed at fostering a comprehensive understanding of the subject matter. Alongside user support activities that focus on specific aspects of the software's functionality, marketing efforts are aimed at the overall purpose. They seek to stimulate the relevant market by clearly and consistently communicating the value of the software application. The choice of marketing activities is primarily influenced by the software's application area and functional purpose. Marketing strategies such as free trials or demos may be effective if the software targets an external end user audience. However, if the application is intended for internal company use, the importance of communication activities should not be underestimated or neglected. In this context, webinars, demonstrations, theme days, or internal incentive systems often prove to be effective.

Overall, the methodology underscores the critical importance of recognizing that all dimensions of market engagement, announced as a software enabler, should not be viewed as a one-time endeavor. Instead, conceptualizing it as a continuous process involving ongoing support, continuous reflection, and sustained promotion is essential.

Practical Guide: ‘Prompt Questions on Market Engagement’

Table 4.8, provides exemplary questions to ensure that the three dimensions—end-user support, feedback loops, and marketing activities—fulfill their function as software enablers under the generic term of market engagement in this methodology for strategic re-engineering.

Table 4.8: Prompt questions on market engagement

Dimension	Relevance	Prompt Questions
End-User Support	Crucial for ensuring smooth adoption and effective use of new software technology.	• Is an onboarding training prepared so that end users know how and for what they should use the software? • Is the training provided on a regular/continuous basis or made available to the end users so that they can access it at any time? • Are instructions provided in different languages and forms (depending on the end-user group)?
Feedback Loops	To continuously improve, optimize and adapt to the needs of end users.	• Is feedback actively collected in all phases (development, testing, implementation)? • Is feedback always collected in the same way or are different techniques used to reach different groups? • What kind of feedback data are collected? • How often is the collected feedback prioritized and finally implemented?
Marketing Activities	To drive awareness, user engagement, and adoption across different target audiences.	• Are target-group-specific campaigns carried out to activate interest? • Is the end-user continuously reminded of the benefits of the software through advertising? • Is the purpose of the intended software implementation spread both internally and externally at different communication levels?

4.6 Evaluation

The evaluation of the entire methodology combines all seven segments from a total of three layers. The main objective of the evaluation is to assess how the current status will be evaluated and to what extent the content of each segment has been considered in the project planning and incorporated into the design, development, and implementation phases. Therefore, the evaluation aims to define effective strategies and practical approaches to prepare software technology for its use in sustainable BMs.

Table 4.9: Levels of maturity, adapted from Sriraman and Raghunathan (2023)

Level	Maturity	Description
1	**Ad hoc**	Lack of consistent approaches, segment is only addressed in a reactive, inconsistent manner.
2	**Reactive**	Segment follows a responsive approach, but without systematic integration.
3	**Proactive**	Segment reaches systematic integration.
4	**Strategic**	Segment is fully integrated, regular performance measurements, and active stakeholder engagements are in place.
5	**Leading**	Segment reaches highest level. It sets benchmarks, shares best practices, and contributes to implement sustainable and digital business models.

For this purpose, a modified version of the CMM method, as presented in Chapter 2.1.2 and widely applied in scientific research, is utilized. This revised version, developed by Sriraman and Raghunathan, redefines the original stages—initial, managed, defined, quantitative, and optimized—into ad hoc, reactive, proactive, strategic, and leading stages. The revised maturity stages offer a practical progression that reflects both current sustainability practices and a roadmap for ongoing improvement, which can be tailored to different organizational contexts within the field of software engineering (Sriraman and Raghunathan, 2023). Table 4.9 explains the meaning of the individual maturity levels before Figure 4.21 presents a visualization of the entire evaluation template for the proposed methodology.

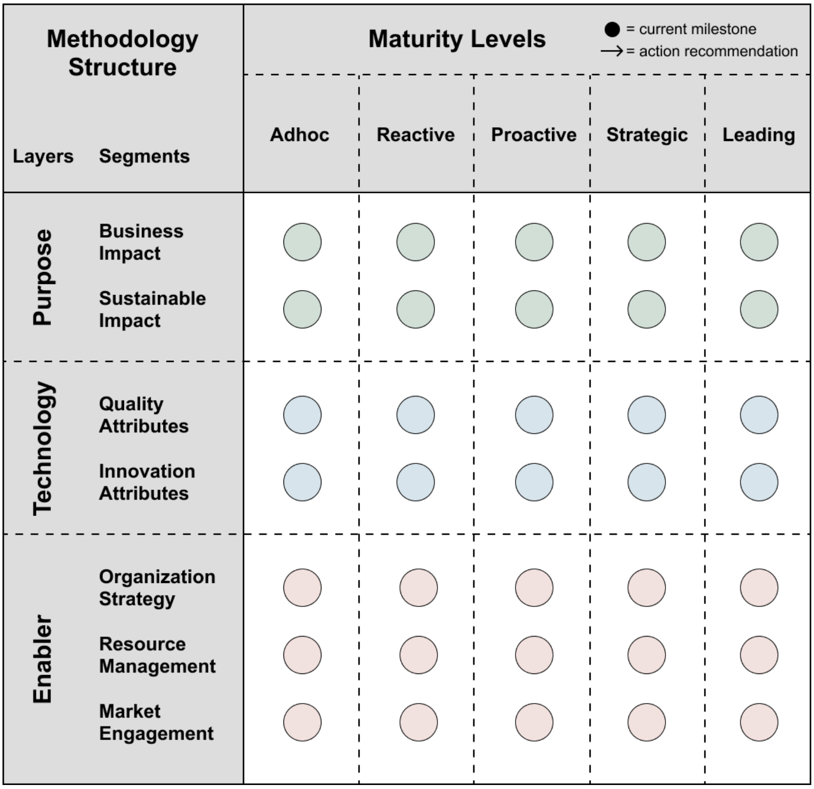


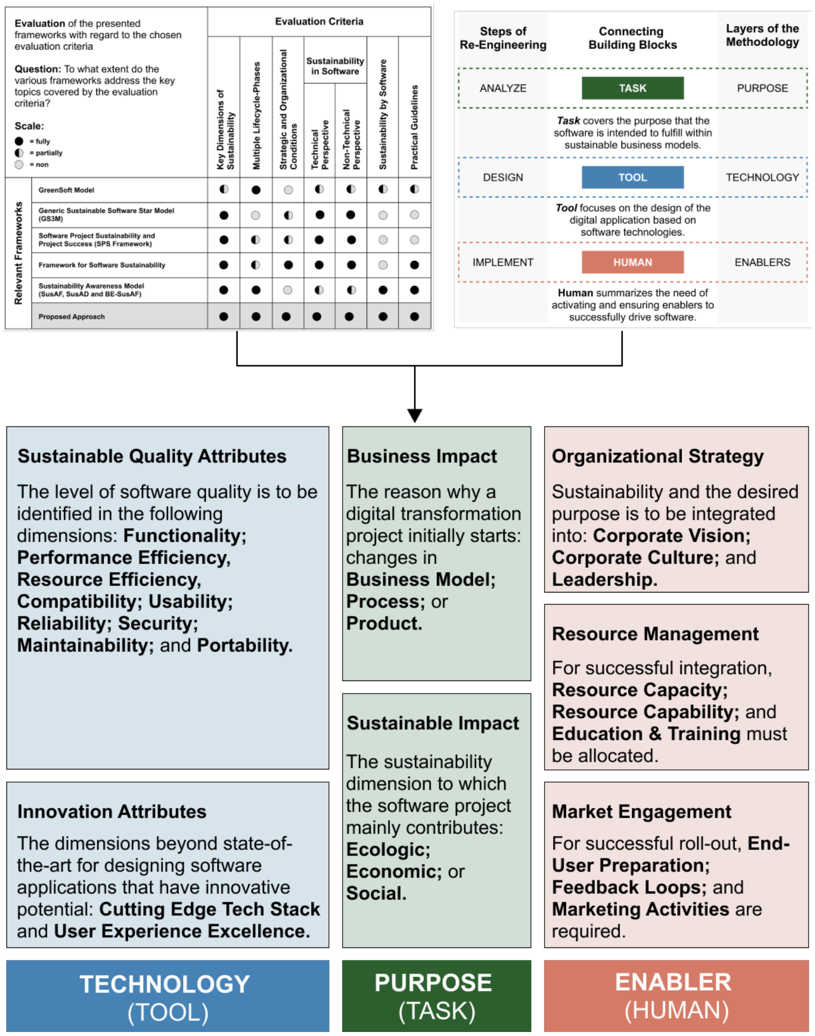
Figure 4.21: Evaluation template for the methodology

The template for the methodology evaluation is structured according to Figure 4.21 by listing the three layers and the individual segments vertically according to the proposed process, while maturity levels are listed horizontally. At the end of the applied methodology, the results from the practical guides suggested after each segment will be used for evaluation across the entire segment. The resulting maturity level is then highlighted with a marker. For this purpose, the filled and color-covered circles are selected in the template. Once the evaluation template is finished, the key recommendations for action should be indicated using vertical arrows. The arrow begins at the current maturity level and directs toward the desired level. The assessment of which action is most important is conducted at the project level and should therefore be tailored to the prevailing environment, where this is done on an individual basis.

The product owner should review and conduct the evaluation at regular intervals. To this end, an exchange with the other roles is useful when discussing various topics and perspectives.

4.7 Conclusion of Chapter 4

The investigation into the state of the art and the associated presentation of five existing frameworks revealed a significant gap in the strategic and organizational integration of software. The lack of systematic consideration for integrating software into existing business processes, which is essential for maintaining long-term relevance and strategic importance, resulted in the development of the methodology presented, as depicted in Figure 4.22.



Evaluation of the presented frameworks with regard to the chosen evaluation criteria

Question: To what extent do the various frameworks address the key topics covered by the evaluation criteria?

Scale:
● = fully
◐ = partially
○ = non

	Evaluation Criteria						
				Sustainability in Software			
Relevant Frameworks	Key Dimensions of Sustainability	Multiple Lifecycle-Phases	Strategic and Organizational Conditions	Technical Perspective	Non-Technical Perspective	Sustainability by Software	Practical Guidelines
GreenSoft Model	◐	●	○	◐	◐	◐	◐
Generic Sustainable Software Star Model (GS3M)	●	○	◐	●	●	○	○
Software Project Sustainability and Project Success (SPS Framework)	●	◐	◐	●	●	○	○
Framework for Software Sustainability	●	◐	●	●	●	○	●
Sustainability Awareness Model (SusAF, SusAD and BE-SusAF)	●	●	○	◐	◐	●	●
Proposed Approach	●	●	●	●	●	●	●

Steps of Re-Engineering	Connecting Building Blocks	Layers of the Methodology
ANALYZE	TASK	PURPOSE
	Task covers the purpose that the software is intended to fulfill within sustainable business models.	
DESIGN	TOOL	TECHNOLOGY
	Tool focuses on the design of the digital application based on software technologies.	
IMPLEMENT	HUMAN	ENABLERS
	Human summarizes the need of activating and ensuring enablers to successfully drive software.	

Sustainable Quality Attributes

The level of software quality is to be identified in the following dimensions: **Functionality; Performance Efficiency, Resource Efficiency, Compatibility; Usability; Reliability; Security; Maintainability;** and **Portability.**

Innovation Attributes

The dimensions beyond state-of-the-art for designing software applications that have innovative potential: **Cutting Edge Tech Stack** and **User Experience Excellence.**

Business Impact

The reason why a digital transformation project initially starts: changes in **Business Model; Process;** or **Product.**

Sustainable Impact

The sustainability dimension to which the software project mainly contributes: **Ecologic; Economic;** or **Social.**

Organizational Strategy

Sustainability and the desired purpose is to be integrated into: **Corporate Vision; Corporate Culture;** and **Leadership.**

Resource Management

For successful integration, **Resource Capacity; Resource Capability;** and **Education & Training** must be allocated.

Market Engagement

For successful roll-out, **End-User Preparation; Feedback Loops;** and **Marketing Activities** are required.

Figure 4.22: Summary of the methodology architecture

The proposed methodology was created based on three layers, seven segments, and 26 dimensions, aiming to develop a holistic approach that should be considered in its entirety. Following the process visualized in

Figure 4.6, the purpose layer, located in the center of the methodology´s architecture, presents the first two segments. Its function lies in determining the business as well as the sustainable impact of the designed software by providing text templates for each dimension to formulate tailored value propositions. Both segments of this layer are designed to address only one dimension, although multiple selections are possible, particularly in the segment of sustainable impact.

The technology layer, marked in blue and located on the architecture's left side, deals with the technological level of the re-engineering process. This layer also comprises two segments: sustainable quality attributes and innovation attributes. While the former is already strongly represented in the existing literature, innovation attributes represent a new segment in this research discipline. Its task is to make the technology intuitive and interesting for the user by incorporating cutting-edge technologies or elements designed for long-term use, such as the utilization of game mechanisms. The latter dimension is thus identified as user experience excellence.

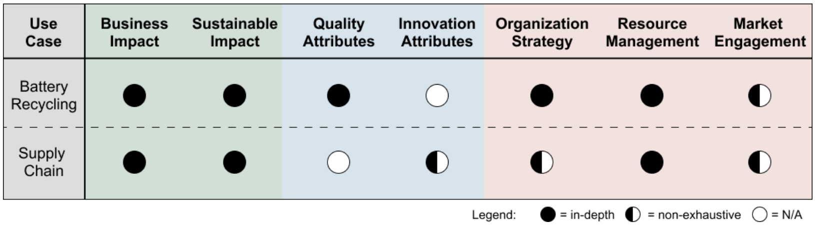
The layer on the right of the designed methodology implies three segments, each with three dimensions. This layer is primarily intended to close the incomplete or missing perspective in the SusAD model. Beyond organizational strategy, this layer deals with the dimensions of resource management and market engagement. The importance of the nine dimensions lies in recognizing that sustainability in software (technology layer) and sustainability by software (purpose layer) are insufficient to actually integrate software and thus achieve its target value at all. Moreover, it is important to consider implementation-oriented dimensions, such as corporate vision, which direct attention to whether the software follows corporate vision. Likewise, the other dimensions, such as resource capability or continuous feedback loops, follow this direction.

One of the key requirements of this methodology is to include practical guidelines (compare RQ3) to serve in a real-world engineering environment. To fulfill this request, each segment includes a practical method for transferring the gained insights into practice. Finally, all the insights gained following the practical guidelines are transferred to the evaluation template in Chapter 4.6. This ensures that each segment receives an assessment of its maturity level across all its dimensions. The aim of this evaluation is to record the current status and to highlight the target status as well as recommendations for action.

Overall, the proposed methodology provides a strategic and global approach, nominating key elements to consider and focus on when working on projects, leading to digital and sustainable transformation.

5 Validation

This chapter presents two projects based on real industry use cases to validate the proposed methodology. All projects were developed and implemented by a software company specialized in developing individual software applications, on behalf of market-leading companies in Germany during 2021–2023. Figure 5.1 provides an overview of which segments of the three layers are covered by each use case and offers initial insights into what extent. The selection of suitable use cases was based on the criteria of whether they fulfilled the characteristics of both digital and sustainable transformation. In addition, the validation use cases focus on topics within the engineering industry. The first case addresses the automotive battery recycling process, while the second use case deals with challenges within the supply chain of an automotive supplier.



Use Case	Business Impact	Sustainable Impact	Quality Attributes	Innovation Attributes	Organization Strategy	Resource Management	Market Engagement
Battery Recycling	●	●	●	○	●	●	◐
Supply Chain	●	●	○	◐	◐	●	◐

Legend: ● = in-depth ◐ = non-exhaustive ○ = N/A

Figure 5.1: Overview of the validation

5.1 Battery Recycling Process

This project focused on the DT of the recycling process for automotive batteries in the chemical industry.

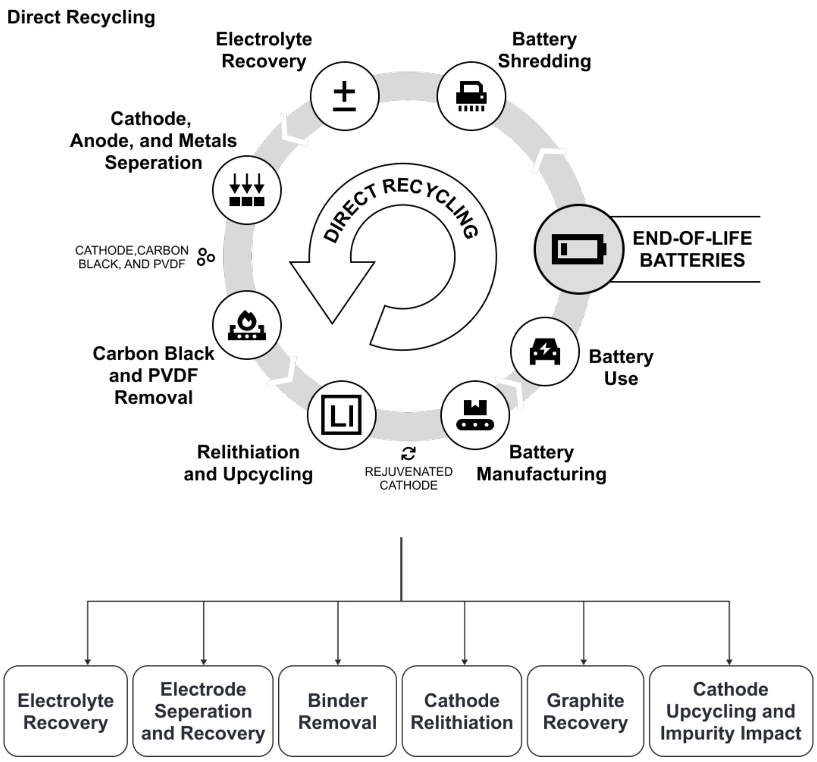


Figure 5.2: Direct recycling schematic diagram, adapted from He et al. (2024)

Since the battery law was effected in 2009, car dealers must take back used batteries. This law also regulates battery distribution, recycling, and environmentally responsible disposal. OEMs and dealers increasingly focus on recycling returned batteries to extend battery life cycles and improve their sustainability footprint. The direct recycling process for lithium-ion batteries is shown as an example in Figure 5.2.

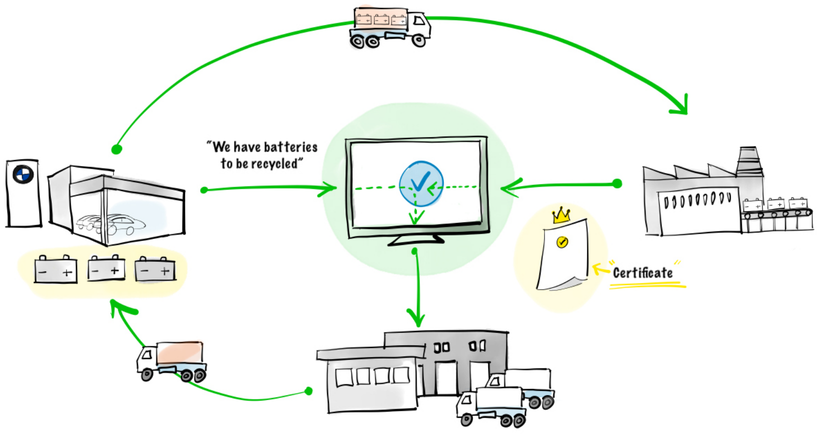


Figure 5.3: Flowchart illustration for battery return

Starting in 2025, new regulations will further require companies to report the carbon footprint of batteries, with 'BatteryPass' becoming mandatory in 2027. By then, at least 50% of lithium used in batteries must be recovered, helping Europe reduce its reliance on rare materials. Beyond simply meeting these requirements, car manufacturers are working to make battery cells more versatile, efficient, and sustainable, advancing both product innovation and their expertise in electric vehicle technology. Chemical companies that produce cathode material, which is crucial for lithium-ion batteries for electric cars, are also interested in this recycling process. The motivational drivers for the recycling of car batteries are ecologically beneficial and strategically important, as Europe relies heavily on Asia for raw materials like nickel, lithium, and cobalt, while importing around 800,000 tons of car batteries annually (Hochwarth, 2024). Both parties, car manufacturers and chemical companies, have therefore joined forces to establish a recycling process.

This process was primarily driven and implemented by the chemical company.

The main focus of this use case is on the central step that deals with the return and traceability of car batteries. This sub-process step should be streamlined and digitized to replace an outdated analog system. Thus, it enables better tracking and data collection, particularly regarding logistics and operational efficiency. A minimum viable product was delivered within the first three months, validating the software's functionality, followed by further enhancements. Overall, the project ensures regulatory compliance and competitive advantage through the fulfillment of the product lifecycle. The resulting battery recycling platform acts as a marketplace where different stakeholders meet, which defines it as a multi-role interaction model (see Figure 5.3). The main stakeholders are the OEM, car dealer, logistic company, and recycling/management company.

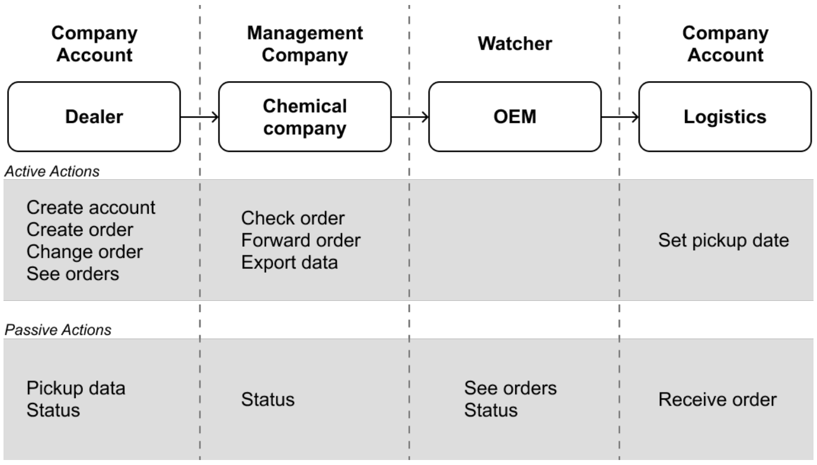


Figure 5.4: Roles and responsibilities within the battery recycling platform

Figure 5.4 outlines the distribution of tasks and responsibilities across roles within the battery recycling platform. Each column represents a specific role, while the corresponding rows detail the actions or tasks assigned to each role. The actions, such as creating orders, checking status, forwarding data, setting pickup dates, and receiving orders, are differentiated according to their active or passive character.

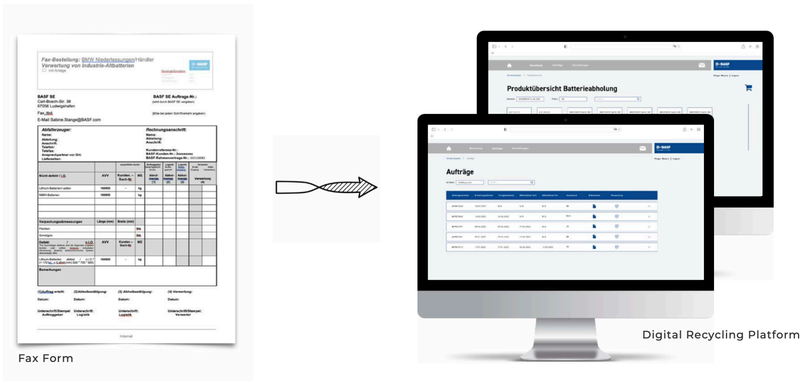
5.1.1 Software Purpose

The goal of the software purpose is to identify the origin value that the individual software application is capable of adding to the company or its end users. Therefore, defining the digital business and sustainable purpose the software aims to fulfill is crucial.

Starting with the digital business value proposition, the developed software solution represents a classic digitization and digitalization project in which an existing analog process is transferred into a digital environment. The process originally involved sending faxes between various stakeholders in the value chain—car dealers, chemical companies, and logistics providers—coordinating the collection and recycling of batteries. Therefore, a process that was previously controlled and operated via fax should be converted into a web solution. Therefore, the following value proposition can be formulated based on the exemplary template provided in Chapter 4.3.1:

The digital business value proposition of the **Battery Recycling Platform** empowers **automotive manufacturers and recycling firms** by seamlessly implementing a new **digital process** that **optimizes the battery recycling process**. Through its **advanced tracking and automation features**, the software addresses **the need for efficiency and transparency** resulting in **reduced operational costs and increased recycling rates**. Unlike **traditional manual processes**, the software developed from the external software partner offers **real-time data analytics as well as significant improvements in process efficiency**.

Therefore, the main purpose of the software application in the business direction lies in the optimization and automation of work processes through the implementation of a digital software environment. A side-by-side comparison of the before and after view is shown in Figure 5.5, where the left side of the visualization shows the fax form as the before version. Concurrently, the result of the digital recycling platform is displayed on the right side.



Fax-Bestellung: BMW Niederlassungen/Händler
Verwertung von Industrie-Altbatterien
☐ mit Anlage

BASF SE
Carl-Bosch-Str. 38
67056 Ludwigshafen
Fax: tbd.
E-Mail: Sabine.Stange@BASF.com

BASF SE Auftrags-Nr.:
(wird durch BASF SE vergeben)
(Bitte bei jedem Schriftverkehr angeben)

Abfallerzeuger:
Name:
Abteilung:
Anschrift:
Telefon:
Telefax:
Ansprechpartner vor Ort:
Lieferzeiten:

Rechnungsanschrift:
Name:
Abteilung:
Anschrift:
Kundenreferenz-Nr.:
BASF-Kunden-Nr.: 2xxxxxxxx
BASF-Rahmenvertrags-Nr.: 202120085

Nicht defekt / i.O.	AVV	Kunden.-Sach-Nr	ME	Abruf-menge (1)	Abhol-menge (2)	Abhol-menge (3)	Verwertung (4)
Lithium-Batterien/-zellen	160605	-	kg				
NiMH-Batterien	160605		kg				
Verpackungsabmessungen	Länge (mm)	Breite (mm)					
Paletten			Stk.				
Sonstiges:			Stk.				
Defekt / n.i.O.	AVV	Kunden-Sach-Nr	ME				
Lithium-Batterien defekt / n.i.O.* (< 170 kg; < (LxBxH mm) 550 * 700 * 500)	160605	-	kg				
Bemerkungen							

(1) Auftrag erteilt: Datum: Unterschrift/Stempel Auftraggeber
(2) Abholbestätigung: Datum: Unterschrift: Logistik
(3) Abholbestätigung: Datum: Unterschrift: Logistik
(4) Verwertung: Datum: Unterschrift/Stempel: Verwerter

Internal

Figure 5.5: Before and after of the application form

Apart from the digital business perspective, the sustainable purpose introduces the second perspective of the software´s value proposition. Therefore, its connection to the ecological dimension becomes evident. While the economic aspect is also addressed through the interconnection of the various dimensions of sustainability, it is not approached directly. The software application applied in this project enables companies to perform, improve, automate, and scale the battery recycling process. The ecological impact is decisive, meaning that this perspective serves as the primary criterion.

Consequently, the following statement summarizes the sustainable impact of the underlying software application:

The sustainability value of the **Battery Recycling Platform** lies in its commitment to the **environmental** dimension. By streamlining and optimizing the battery recycling process, the software **reduces waste, reuses resources, and improves energy efficiency** resulting in a **lower carbon footprint**. This software technology supports companies in applying circular economy practices.

The 'Battery Recycling Platform' use case defines both a digital business-oriented and a sustainability-oriented value proposition. This fulfills the desired purpose of the software technology.

5.1.2 Software Design

The second layer focuses on the design of the software technology, starting with the quality attributes.

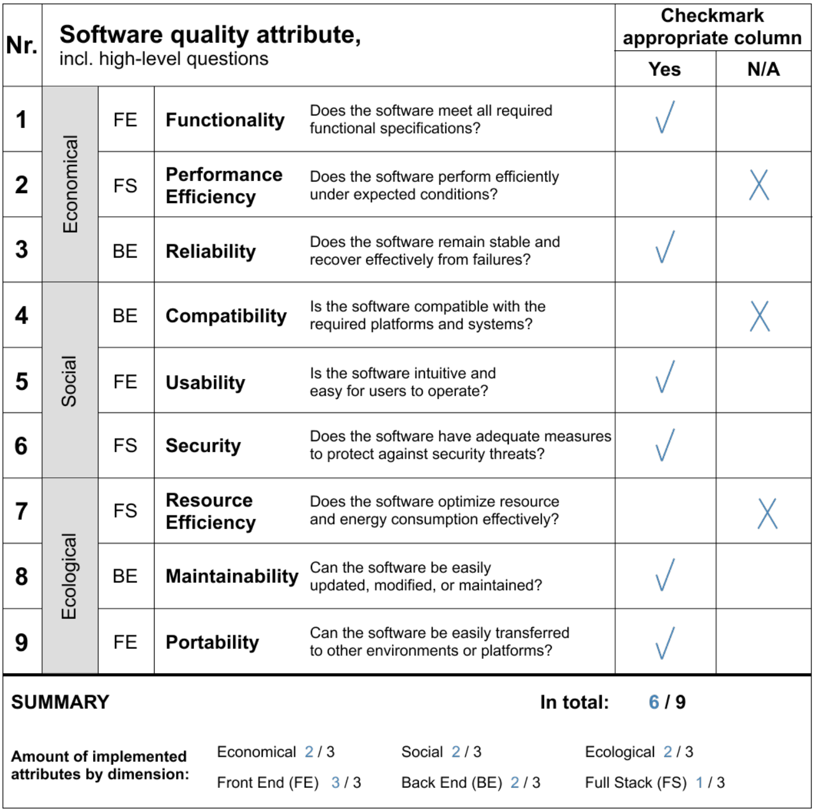
To incorporate the nine quality attributes of sustainable software mentioned in the methodology into the battery recycling platform to the highest extent, two key decisions are made. First, the team selects a cloud-based structure for hosting, servers and databases to implement the aforementioned software. This should serve to meet high global security standards and positively influence system and reliability. In addition, this cloud-based environment, which is represented by Amazon Web Services, for example, enables the fast and efficient provision of application workloads with low latency times. Second, a so-called three-tier design is chosen for the infrastructure of the recycling platform, which divides the infrastructure into one public and two private tiers. The main advantage of this variation is the seamless and flexible scaling of performance and storage space. As a cloud-based web application, the software can therefore be classified as highly portable, which is also due to the technological focus on scalability. In addition, each level of the architecture can be extended through horizontal scaling and load balancing to efficiently handle increasing traffic and requests.

Regarding the maintainability attribute, the main focus is on modularity as a sub-attribute. The underlying decision to use the three-tier architecture has the advantage that the recycling platform can be modularized individually, which means that each part can be managed independently of the other parts moving forward. Therefore, changes can be made very quickly, which in turn reduces resource requirements.

To ensure a high level of reliability, major attention is paid to high fault tolerance during software development. This enables the system to remain in operation, even if some of the components used to build the system fail. This performance is guaranteed by maintaining a redundant system.

The security of a digital system is fundamental at all stages of development and is essential for the long-term, sustainable use of the application. As part of the battery recycling project, several standard security measures will be implemented, including secure login procedures and data encryption, to protect the system from attacks and ensure data integrity.

For secure login, SSL is used to establish encrypted connections and ensure that sensitive login data cannot be intercepted. In addition, token-based authentication is implemented to validate user sessions, with unique tokens assigned to each user to prevent unauthorized access. Another aspect is the authentication of actions through a role-based control system, in short RBCS, which enforces restrictions on user actions based on predefined roles. Moreover, passwords are hashed with salt values, which should make reverse engineering or cracking password data computationally difficult. Several additional measures, from regular backups to data encryption, are also implemented to improve the protection of user data and meet the high security requirements of sustainable software.



Nr.	Software quality attribute, incl. high-level questions				Checkmark appropriate column: Yes	Checkmark appropriate column: N/A
1	Economical	FE	**Functionality**	Does the software meet all required functional specifications?	√	
2	Economical	FS	**Performance Efficiency**	Does the software perform efficiently under expected conditions?		X
3	Economical	BE	**Reliability**	Does the software remain stable and recover effectively from failures?	√	
4	Social	BE	**Compatibility**	Is the software compatible with the required platforms and systems?		X
5	Social	FE	**Usability**	Is the software intuitive and easy for users to operate?	√	
6	Social	FS	**Security**	Does the software have adequate measures to protect against security threats?	√	
7	Ecological	FS	**Resource Efficiency**	Does the software optimize resource and energy consumption effectively?		X
8	Ecological	BE	**Maintainability**	Can the software be easily updated, modified, or maintained?	√	
9	Ecological	FE	**Portability**	Can the software be easily transferred to other environments or platforms?	√	

SUMMARY		**In total:**	6 / 9
Amount of implemented attributes by dimension:	Economical 2 / 3	Social 2 / 3	Ecological 2 / 3
	Front End (FE) 3 / 3	Back End (BE) 2 / 3	Full Stack (FS) 1 / 3

Figure 5.6: Completed checklist of software quality attributes

Regarding the attributes of functionality, usability, and compatibility, no specific measures were emphasized during software development to achieve particular standards. Testing confirmed that functionality and usability met the necessary requirements, with the platform operating as intended and usability proving straightforward and intuitive for users. However, no information was available regarding compatibility.

Figure 5.6 provides the completed checklist as provided within the methodology.

The second segment in the design layer encourages the integration of innovative design elements. Neither cutting-edge technologies nor advanced user experience elements such as gamification or other UX excellence components are used in implementing the battery recycling platform. Hence, the focus remains on functionality and efficiency and not on the integration of novel design paradigms or user engagement strategies. Consequently, no significant contribution to sustainability will be made from this point of view.

Looking at the sustainable quality features in this use case, 2 out of 3 features were actively applied in all three sustainable dimensions. The social dimension was only elaborated in detail in the area of safety, while end users tested the usability attribute. The relevance of the compatibility attribute is questioned, as at the current time of the evaluation, only a closed group of people uses the software. However, should the management company decide to scale the platform for other raw materials, an examination of the social attributes is highly recommended to ensure a long-term and satisfying use of the platform.

In addition, further development of the platform seems to be highly interesting due to the relevance of the purpose it addresses. In this regard, checking the feasibility of suitable innovation attributes for the software is highly advisable. This will not only improve the user experience but also unlock new potential through the application of latest technologies, ensuring that the platform stays updated.

5.1.3 Software Enablers

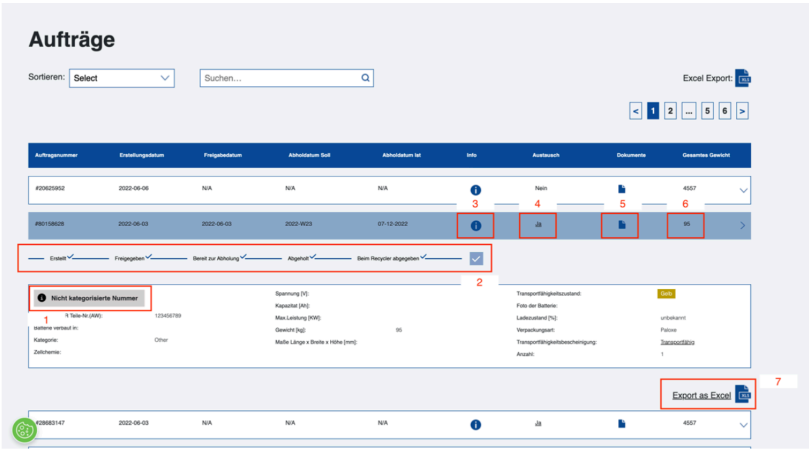
As the successful implementation of the battery recycling application is primarily dependent on the active involvement of the dealer and the management company, both parties are nominated for the software enabler layer. Although the role of the logistics company is important overall, its function is transferable and is only required on demand, resulting in a support function.

Since digitization and sustainability are anchored in the management company's vision, raising awareness internally about the importance and necessity of this project seemed unnecessary. The company sees itself as a reliable solution provider along the entire value chain of the battery life cycle. With this in mind, the introduced software project contributes markedly to the company's vision. By organizing action days and internal training courses and by carrying out several projects in this area, the corporate vision is already equally anchored in the corporate culture.

While all three dimensions of the organizational strategy are implemented and lived in an exemplary manner at the management company, which is the main driver of the recycling process, the role of the car dealer shows a deficit. As the corporate vision is not anchored in the individual car dealer but is defined and communicated by the OEM, which in turn is responsible for establishing the recycling tool, the vision can be declared to exist. However, leadership and culture are individually dependent on the dealer, which makes a general assessment difficult with more than 500 connected car dealers.

The management company seemed to have sufficient resource capacity available throughout the entire project cycle. Despite an initial setback, as the project could not be realized through the provider Shopify, both human and financial resources remained adequate to build the project within a customized software environment. Moreover, a dedicated team comprising diverse competencies was assembled specifically to ensure project success. This team could deliver information in a timely manner, ensuring a smooth development phase, while maintaining excellent communication within such a cross-functional team.

The resource management of the car dealer can be deemed appropriate and sufficient without further analysis, as from the perspective of the necessary resources for utilizing the software environment, no additional resources are required compared to the analogous process. The required capacity remains constant, if not lower, and the resource competence shows no significant deviation.


Aufträge

Sortieren: Select — Suchen... — Excel Export:

< 1 2 ... 5 6 >

Auftragsnummer	Erstellungsdatum	Freigabedatum	Abholdatum Soll	Abholdatum Ist	Info	Austausch	Dokumente	Gesamtes Gewicht
#20625952	2022-06-06	N/A	N/A	N/A	●	Nein	▮	4557
#80158628	2022-06-03	2022-06-03	2022-W23	07-12-2022	● 3	Ja 4	▮ 5	95 6

Erstellt ✓ — Freigegeben ✓ — Bereit zur Abholung ✓ — Abgeholt ✓ — Beim Recycler abgegeben ✓ — ✓ 2

Nicht kategorisierte Nummer 1

R Teile-Nr.(AW):	123456789	Spannung [V]:		Transportfähigkeitszustand:	Gelb
Batterie verbaut in:		Kapazitat [Ah]:		Foto der Batterie:	
Kategorie:	Other	Max.Leistung [KW]:		Ladezustand [%]:	unbekannt
Zellchemie:		Gewicht [kg]:	95	Verpackungsart:	Paloxe
		Maße Länge x Breite x Höhe [mm]:		Transportfähigkeitsbescheinigung:	Transportfähig
				Anzahl:	1

Export as Excel 7

Auftragsnummer	Erstellungsdatum	Freigabedatum	Abholdatum Soll	Abholdatum Ist	Info	Austausch	Dokumente	Gesamtes Gewicht
#28683147	2022-06-03	N/A	N/A	N/A	●	Ja	▮	4557

Figure 5.7: Excerpt from the instruction documentation

With regard to the third dimension, education and training of resources, the management company offers and regularly conducts in-house training to raise awareness and further educate on the topic of cyber security. In addition, training instructions have been created, developed and distributed in several languages and tailored to the various stakeholders involved in the process. This is intended to ensure a smooth onboarding process and error-free usage of the software. Figure 5.7

shows an excerpt from the instruction documentation on how to interpret and use the functions in the software.

For the management company, the end-user support dimension only partially applies, as no active participation existed during the development phase and this party took over the complete design of the digital application. End-user support was given here in the sense of technical support, which the external software company provided to the management company, even after the software had been fully migrated to the management company's own environment.

However, active end-user support is important for car dealers in order to ensure that the recycling platform is adopted, accepted and used regularly. From the start, the challenge was to convince the individual car dealerships of the benefits and added value of the digital platform. Nevertheless, the fear of additional work and a complex digital environment was quickly overcome by organizing training sessions, digital presentations and providing step-by-step instructions and explanatory videos. As a result, the end user from the car dealer side was also convinced of the work guidance and acceleration provided by the software application.

During the entire development phase, significant attention is paid to gathering regular feedback from different end users to ensure that the designed software meets the actual needs and expectations of the users. This also frequently provides new insights into the actual workflow, allowing the system to be designed in a more practical and realistic way. For this use case, Table 5.1 displays a selection of received feedback and proposed improvements, communicated through a standard Excel spreadsheet.

Table 5.1: Anonymized examples from the feedback spreadsheet

Category	Actual Situation - Description	Target Situation - Description	Date	Name	Operating System	Browser	Version
Enhancement	Upload of images only works in pdf. not in png. format	Upload should work in all standard formats	27.01.22	Max Muster	Windows 10	Firefox	99.0.1
Bug	Pick-up date entered, status does not change	Status should change according to the process	27.01.22	Max Muster	Windows 10	Firefox	99.0.1
Bug	Requested pick-up date cannot be entered if a pick-up date is already filled in	A request for a preferred date should be possible	27.01.22	Max Muster	Windows 10	Firefox	99.0.1
Enhancement	Search field does not work with customer data	Search field should also work for customer data such as location.	27.01.22	Max Muster	Windows 10	Firefox	99.0.1

To effectively implement the platform and integrate it among the stakeholders, several software enablers were identified. Up to this point, the robust structure of resource management and the many actions taken to support end users in the segment of market engagement have been crucial. Overall, the landscape of software enablers can be seen as an important positive factor for the success and sustainable integration of software.

Various software enablers were identified to implement the platform appropriately and to integrate it among different parties. So far, the decisive factor is the strong structure of resource management and the multiple actions for end-user support within the dimension of market engagement. Overall, the software enabler landscape has a positive influence on the implementation of a sustainable software.

5.1.4 Fazit

The complexity of the project was moderate due to the cognitive and technical simplicity of the task—digitizing an existing analog process. However, the involvement of multiple stakeholders, including automotive dealerships, logistics providers, recycling companies, and chemical industry players, added a layer of complexity. Ensuring that each party had a customized user interface that integrated smoothly with the others required careful planning and execution. Every user journey was meticulously designed to ensure the correct flow of commands and data.

Careful implementation and transition-out led to the success of the project. From the management company's perspective, all expectations were met, goals were achieved and sovereignty over the tool was secured.

The application of the proposed three-layered methodology has led to the identification of not only a robust digital business and a sustainable software purpose, but also a solid and scalable software technology design, along with key software enablers. As a result, it can currently be assumed that the battery recycling platform will be utilized in a long-term, continuous, and function-oriented manner. Furthermore, the architecture, which possesses a ready-to-scale character, makes it feasible to extend the platform to other recyclable raw materials, thereby establishing it within a broader market. This further underscores the significance of the software in supporting sustainable BMs.

As shown in Figure 5.8 the completed evaluation for this use case, it is the responsibility of leadership and business analysts to expand the business purpose toward new horizons—for example, leveraging the platform to create added value from collected data and information. For this, following the first recommendation and updating the software with innovation attributes is useful. This could involve introducing new capabilities through cutting-edge technologies, resulting in predictions, optimizing routes, or recognizing patterns within systems-of-systems, thereby driving value creation through digitalization. Moreover, concerning the second recommendation, an iterative feedback process should be considered, as well as further marketing activities to onboard more car dealers or open discussions to other industries with similar issues and needs.

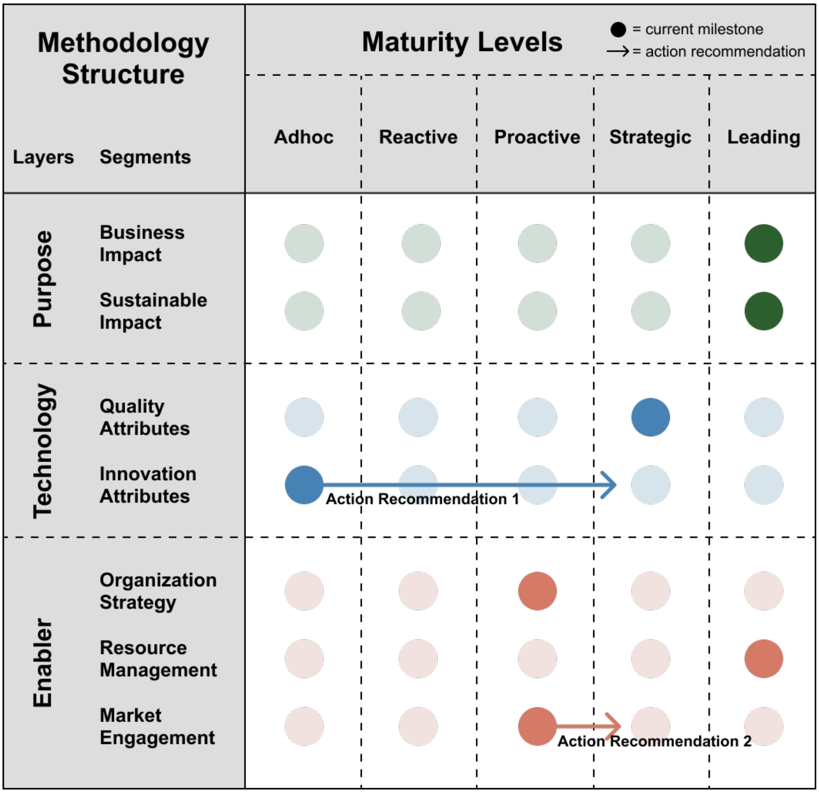


Figure 5.8: Completed evaluation for battery recycling use case

5.2 Supply Chain Management

The second use case, likewise from the automotive industry, is about customized software along the supply chain. The idea behind the software concept was developed as part of a multi-day workshop on the digitization of processes in connection with the overarching goal of getting closer to the customer. Establishing a closer or more direct customer relationship is expected to increase informative data as a basis for improving operational, ordering, and delivery processes.

The company faces significant additional costs within the supply chain. The reason lies mainly in frequent last-minute order changes (up to 30%–40% of the order volume) made up to four weeks before delivery. This behavior, combined with the lack of insights, results in significantly increased special shipments, which have a notable impact on the company's sustainable footprint. Consequently, a mid-double-digit million-Euro amount arises from short-term changes in customer orders and manual allocation practices within the company. Therefore, the overarching goal of this project is to enhance customer interaction and improve sales performance by streamlining these processes.

The defined use case focuses on optimizing the delivery process by integrating customized software between the company and its customers. The resulting multi-role interaction model comprises three different roles. The role setup includes customer solution, team members in manufacturing, and procurement.

The software is designed as a digital solution to markedly reduce the high volume of special shipments by empowering individual stakeholders in the process with greater decision-making authority. Concurrently, it actively reinforces the associated responsibilities from both a business and sustainability perspective, fostering more informed and accountable decision making. The gamified logic of the software reflects the underlying magic to motivate all stakeholders to participate and act wisely by simultaneously reducing complexity.

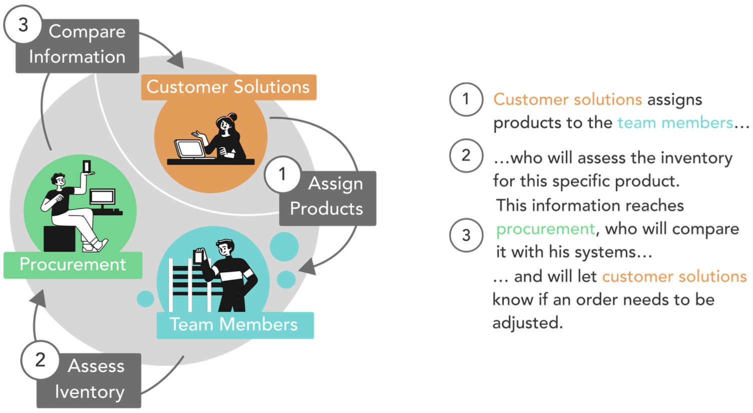

3 Compare Information
Customer Solutions
1 Assign Products
Procurement
Team Members
2 Assess Iventory
1 Customer solutions assigns products to the team members...
2 ...who will assess the inventory for this specific product.
This information reaches
3 procurement, who will compare it with his systems...
... and will let customer solutions know if an order needs to be adjusted.

Figure 5.9: Overview of the core flow of the second use case

Before applying the methodology of this to the current use case in the following sub-chapters, Figure 5.9 illustrates the defined process for which the application has been designed. The tasks and responsibilities are assigned to the individual roles. For example, the customer service department has the task of nominating certain orders in the application that the customer should check. Once the employee in the manufacturing department assesses how many products of this order are still in stock and how long the stock will last, the purchasing department compares the information with the available data and verifies it. Finally, the procurement department sends feedback to customer service. The architectural structure of the software application is provided in Figure 5.10 for more information.

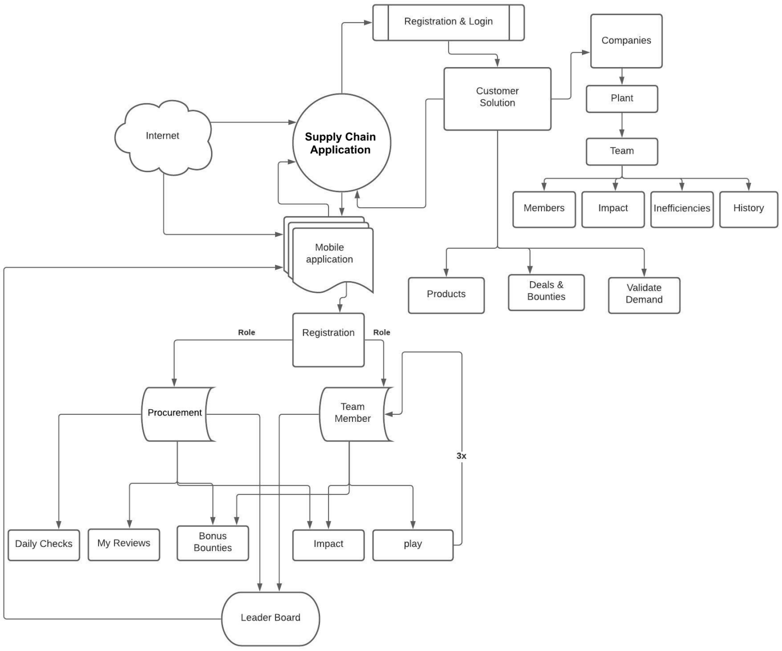


Figure 5.10: System architectural diagram of the supply chain use case

5.2.1 Software Purpose

The primary business purpose of this software has two dimensions. First, the software, designed as a collaborative platform with multiple functions as shown in a screenshot in Figure 5.11, aims to improve transparency and communication between the involved parties, in order to strengthen customer relationships and improve overall interaction.

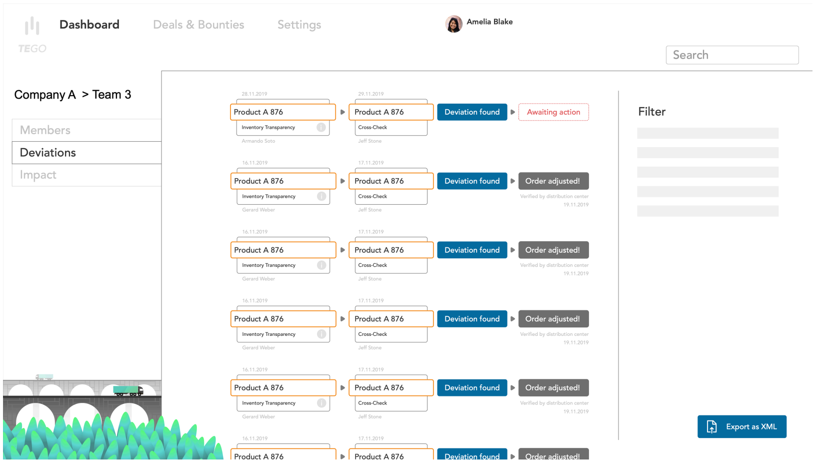


Figure 5.11: Exemplary screenshot of the customer solution interface

Second, the software offers a new digital touchpoint between the company and its clients, which is designed to reduce the number of special shipments. This in turn saves money and time and reduces operational complexity, ultimately leading to better sales results. Moreover, this approach follows the opinion that information systems enhance an organization's ability to connect and support its members, create structured knowledge resources, and strengthen its capacity to bridge gaps across different areas or locations (Dewett and Jones, 2001).

According to the predefined template, the business purpose of the software can be described as follows:

The digital business value proposition of **Supply Chain Software** is to empower the **automotive supplier and its customers** by seamlessly implementing a new digital process that **improves transparency and communication between those parties.** Through its **gamified interface and integrated communication tools**, the software addresses the need **to strengthen customer relationships and to improve interactions, as well as reduce special shipments, resulting in cost savings and time efficiency**.

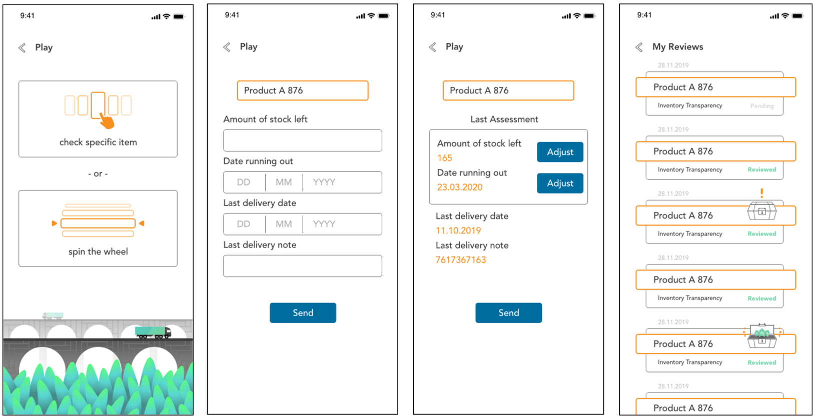
To complete the software purpose, a further step is to define the sustainable impact. For the second segment of the first layer, the methodology also provides a template for direct transfer to the practical context. Based on the previous description of the software, the following sustainable value proposition can be derived:

The sustainability value of **Supply Chain Software** lies in its commitment to the **ecologic** dimension. By implementing **gamified data and communication flows**, the software contributes to **achieving improved ordering processes and reduced transportation volumes**, resulting in a **reduction in emissions**. This software supports companies in **controlling special shipments to lower costs for all parties involved and to reduce the corporate carbon footprint.**

The Supply Chain use case articulates a value proposition that encompasses both digital business objectives and sustainability goals, thereby establishing a comprehensive purpose for the software.

5.2.2 Software Design

As illustrated in Figure 5.1 in the introduction to this chapter, this use case lacks validation for the quality attributes segment within the software technology layer since the project was closed before it was completed. Although the quality of software technology cannot be validated for its sustainable utilization, the level of innovation in the software can be examined more closely. Consequently, the segment about innovation attributes will be analyzed further in terms of user experience excellence. In this context, the gamification approach used within software design, as exemplified in Figure 5.12, has a positive impact on user experience, hence increasing the likelihood of continuous and long-term use. Figure 5.13 illustrates the gamification profile of this software application, highlighting the game mechanics used to address the eight core drives of human motivation within this software design.



Amount of stock left
165
Date running out
23.03.2020
Last delivery date
11.10.2019
Last delivery note
7617367163

28.11.2019
Product A 876
Inventory Transparency — Pending
28.11.2019
Product A 876
Inventory Transparency — Reviewed
28.11.2019
Product A 876
Inventory Transparency — Reviewed
28.11.2019
Product A 876
Inventory Transparency — Reviewed
28.11.2019
Product A 876
Inventory Transparency — Reviewed
28.11.2019
Product A 876

Figure 5.12: Exemplary designs for mobile application

Given that the software was discontinued, this relationship can only be inferred from impressions gathered during the two testing phases. The first testing phase, a closed alpha phase, was done internally, while the second phase, called open beta, was executed with external stakeholders and within an international audience. The specifics of the gamification context will not be detailed here, as the application of gamification elements itself is essential for validating the innovation attribute of the software. Consequently, the playful elements and mechanisms of the software application could be documented as an innovation attribute during these test phases, as it fostered motivation, curiosity, and engagement among the end users. According to the Kano model, gamification can be categorized as an excitement attribute in the UX landscape that contributes to user experience excellence. It can, therefore, be identified as an innovation attribute within software technology.

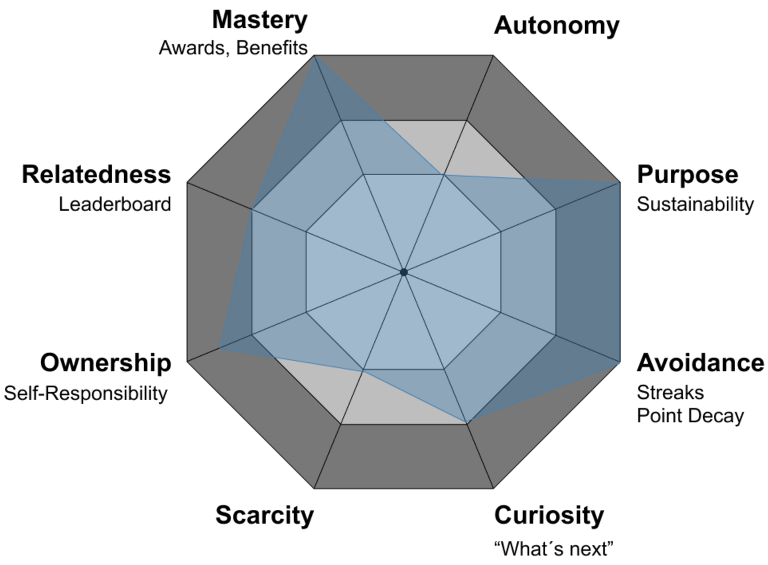


Figure 5.13: Gamification profile of use case 2

From the innovation attribute point of view, the software technology is well prepared for engaging its user on a long run. The gamification approach adds an excitement level not only to the application itself but also to the whole process workflow. It fosters team dynamics and introduces a touch of competition to motivate higher accuracy within the supply chain processes.

5.2.3 Software Enablers

The software enablers layer was validated across three defined segments: organizational strategy, resource management, and market engagement.

Organizational Strategy

Even if the purpose of the software could be defined on both the business and sustainability dimensions, no alignment with the company's overall strategy exists during the software design. The software purpose addressed was derived from a standalone problem. No alignment with other topics addressed by the company was established to ensure the effort of developing an in-house software solution had a comprehensive impact. Furthermore, although the topic of sustainable BMs was known, it had neither been considered in the company's own BMs and processes nor prioritized in processes. The defined purpose was therefore either too narrowly focused on this one pain point, or the topic of sustainability was not actively embedded in the vision of the company's day-to-day operations (until the end of the project).

The other relevant dimension in the area of organizational strategy comprises the existing leadership in the project. In this case, it has to be fulfilled by the Supply Chain Manager (or higher) and the responsible Product Owner. These tasks were mainly covered in the first half of the project but gradually became increasingly fragmented. Personnel changes during the course of the project resulted in leadership competencies being partially or entirely lacking by the project´s conclusion.

Despite the challenge posed by the partial lack of leadership, the remaining project team generated a slight euphoria and, above all, curiosity within the company itself in connection with the gamified approach in the software technology layer. However, this positive expectation could not survive external economic challenges. These external factors prevailing at the beginning of 2022, such as the start of the Ukraine War and the ongoing global pandemic, made international customer relationships more difficult, resulting in little willingness and acceptance of the introduction of a new gamified software solution to improve processes and reduce emissions. Overall, the culture in the environment foreseen for the software could be classified as difficult, with little willingness and understanding for change.

Resource Management

At the beginning of the software project, the resource capacity and capability for this use case met the minimum requirements necessary for project execution. Human resources are sufficiently available for realization, with team member availability aligning adequately with the project timeline and milestones. While an infrastructure upgrade was previously required, management promptly approved and released it, ensuring that technical resources remained sufficient. However, following the test phase, a shortage of personnel resources emerged due to further staffing changes, specifically the absence of a product owner.

Unfortunately, no information exists for evaluating the resource education and training dimension.

Market Engagement

End-user support is essential for facilitating smooth adoption and effective use of the software. Onboarding materials, including detailed explanations of both the software's usage and intended purpose, were prepared to guide end users. Training and instructional materials were developed prior to the test phase and further refined based on user feedback. These resources were available in multiple languages and formats to meet the needs of different user groups.

However, due to the lack of a full-scale rollout, a comprehensive evaluation cannot be provided. Therefore, this assessment is based on user reactions and behavior observed during the test phases, which indicated positive engagement and understanding of the software's functions. End-user interactions during testing suggest that the support materials were well received and effective in meeting users' needs. Further evaluation would be required following a complete deployment to confirm long-term effectiveness and satisfaction with the end-user support structure.

In this use case, feedback was actively collected during each project phase, from development through testing. However, feedback was only gathered manually, which presented challenges due to the diverse user roles, the large number of features at an early stage, and the variety of end-user devices (iOS, Android, and web). An example of manually collected feedback is shown in Table 5.2. This complexity makes it difficult to manage and implement feedback consistently. In addition, the absence of a prioritization process led to numerous unnecessary changes and adaptions before the initial rollout. Overall, the influencing factors were numerous at the early stage.

Table 5.2: Anonymized examples of the collected feedback

Category	Actual Situation - Description	Target Situation - Description	Date	Name	Operating System	Browser	Version
Bug	Procurement: wrong late acitivity time is shown	Latest activity time should be shown	27.01.22	Max Muster	Windows 10	Firefox	99.0.1
Bug	Notification is not working correctly	Notification should be displayed when there are new updates	27.01.22	Max Muster	Windows 10	Firefox	99.0.1
Bug	Button "adjust" does not accept updated numbers	After clicking "adjust" it should be possible to change the input and save it	27.01.22	Max Muster	Windows 10	Firefox	99.0.1

The initial marketing efforts included an explanatory video and an internal promotional video, which generated initial excitement and user engagement for a certain period. However, due to the project's early termination, no further marketing activities were pursued.

This use case revealed significant gaps across the entire Software Enabler layer. None of the nine dimensions demonstrated a satisfactory outcome throughout the design and development phases. Beyond the absence of a product owner, critical deficits were found in the organizational strategy segment, particularly in alignment with the vision, support from leadership, and agile culture.

5.2.4 Fazit

Figure 5.14 shows the results of the evaluation. Despite the strong initial value proposition with a dedicated purpose regarding sustainability, this use case demonstrates that software for sustainable BMs must sufficiently address all three layers of the methodology. A targeted value proposition that addresses both business and sustainability goals, even when combined with innovation attributes, is insufficient to ensure the use of a software application within a sustainable BM.

Moreover, it confirms that software technology is an intensely human-centered activity. Humans are responsible for developing software, making resource-related decisions, and, most importantly, using software applications. This insight suggests that, in such cases, the software enabler layer should be improved or added to increase the potential for added value in return for the effort invested.

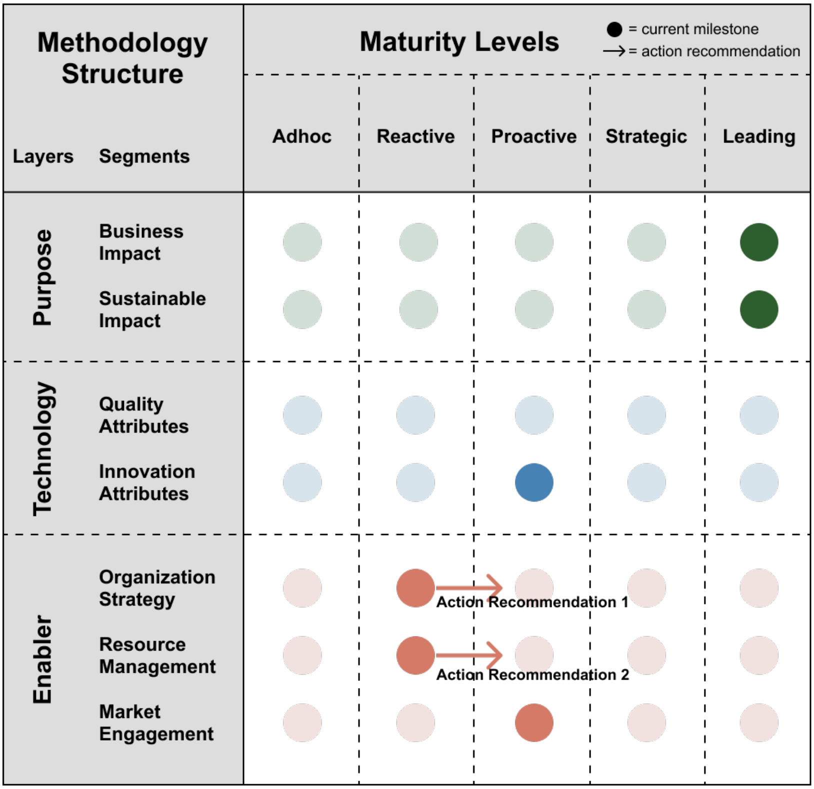


Figure 5.14: Completed evaluation for the supply chain use case

Overall, the aforementioned use case illustrates the necessity of maintaining a consistent and reliable supply of resources and leadership expertise, beyond ensuring unwavering alignment with the corporate vision and fostering an agile mindset within an open organizational culture.

5.3 Conclusion of Chapter 5

Before summarizing Chapter 5, this thesis remark that both use cases conducted and presented relate to the automotive industry in Germany. Both use cases address contemporary pain points, which are explained in the first layer, software purpose. The software is fully functional in the first example which enables the analysis of quality attributes. Unlike the first use case, the second use case did not include an analysis of the quality attributes; however, assessing the innovation attributes is possible. Accordingly, all dimensions of the second layer could also be validated.

In addition, the validation highlighted the importance of the third layer, which has not been considered in detail or at all in previous methods and frameworks, as described in Chapter 3.3. This layer significantly influences whether a software application can be successfully implemented in the long term, thereby enabling a targeted contribution to sustainable BMs. Although the basic concept of use case 2 was valid and up-to-date and was also categorized as valuable by the company, the implementation failed due to factors in the third layer, software enablers.

Extensive practical cases should be conducted to further investigate these conclusions and thus confirm the robustness of the presented methodology. Since the scope of this study is limited, further research efforts are necessary to generate sufficient data and enable a longer-term analysis.

In total, the methodology validation includes both a successful example and a negative outcome, as use case two was unsuccessful. By directly identifying potential improvements, the concept of the methodology, the creation of an approach that re-engineers software technology for its use in sustainable business models, can be confirmed.

6 Discussion

This chapter discusses the main findings of the thesis and includes two main sections. Section 6.1 reiterates the research problem, objective, and derived research questions. Section 6.2 acknowledges the limitations of the research and provides suggestions for future research.

6.1 Revisiting the Research Questions

The declared objective of this thesis is to create a methodology that aims to strategically design software technologies in a way that they add value for and in sustainable BMs. This goal is linked to the complexity of the twin transformation, which deals with digital and sustainable development. In addition, the problem of the high error rate in DT projects is initially emphasized, as is the often insufficient use of implemented individual software solutions.

To demonstrate how the thesis has met its original objectives, the three specified research questions are summarized and answered below. Figure 6.1 provides insights regarding which (sub)chapters are assigned as answers to which research question.

Research Question 1: How can software technology that balances digital and sustainable transformation be strategically re-engineered?

This question was addressed in two sequences. First, the three steps of a re-engineering process were projected onto the topic of software development, which led to the derivation of the so-called building blocks. The analysis was then transferred to the software task, the design to the software tool and the implementation to the person in charge. Second, by combining the steps with the building blocks, the fundamental pillars for the strategic re-engineering of software technologies in the field of digital and sustainable transformation were established. The core pillars include the determination of the purpose, the design of the technology and the implementation of enablers. Consequently, the research question can be answered by arguing that this goal can only be successfully achieved through a holistic perspective of the purpose, the technology, and the enablers. These pillars represent the core structure of the methodology and are labeled as layers throughout the thesis. The results are summarized in Figure 4.2. Therefore, subchapter 4.2.2 primarily contributes to answering the first research question.

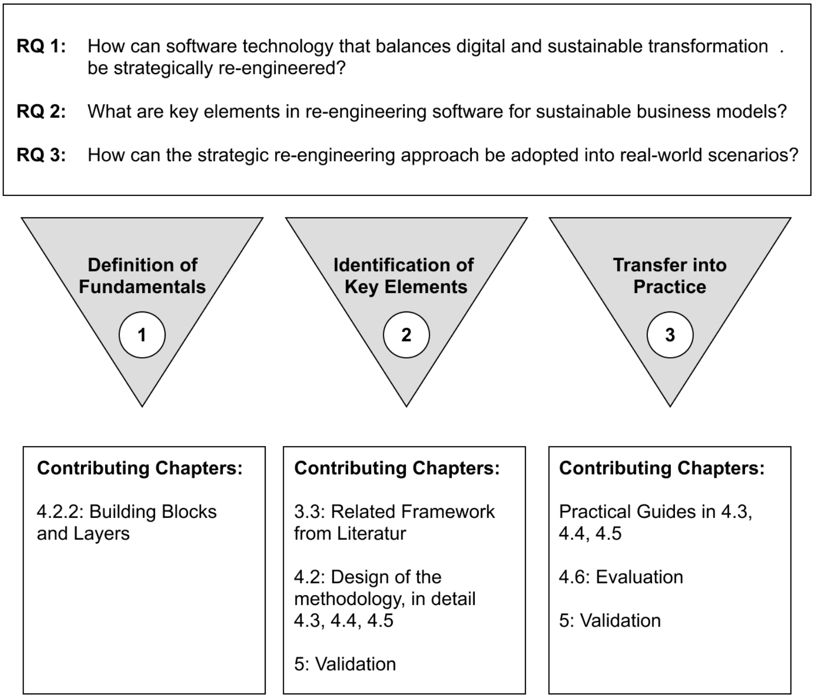

RQ 1: How can software technology that balances digital and sustainable transformation . be strategically re-engineered?
RQ 2: What are key elements in re-engineering software for sustainable business models?
RQ 3: How can the strategic re-engineering approach be adopted into real-world scenarios?
Definition of Fundamentals
1
Identification of Key Elements
2
Transfer into Practice
3
Contributing Chapters:
4.2.2: Building Blocks and Layers
Contributing Chapters:
3.3: Related Framework from Literatur
4.2: Design of the methodology, in detail 4.3, 4.4, 4.5
5: Validation
Contributing Chapters:
Practical Guides in 4.3, 4.4, 4.5
4.6: Evaluation
5: Validation

Figure 6.1: Contribution of chapter according to research questions

Research Question 2: What are the key elements in re-engineering software for sustainable BMs?

Based on the literature research and the assessment of various models conducted in Chapter 3.3, re-engineering for sustainable BMs should extend beyond the technical and economic levels, requiring strategic and organizational integration to ensure comprehensive implementation. This gap was identified by comparing the models in Table 3.1. Accordingly, and considering the results from the first research question, the technical level was assigned to the technology layer, the economic level, represented by the value proposition, mainly to the purpose layer, and the strategic and organizational level to the enabler layer.

These three layers were subdivided into seven further areas in Chapter 4.2. The purpose layer thus revealed the business and sustainability impact, the technology layer the quality and innovation attributes and the enabler layer the organizational strategy, resource management and market engagement. All of these areas, referred to as segments, represent key elements of the scope under investigation. For a better understanding of the scope of the seven key elements, they were subdivided into individual dimensions (see Figure 4.4). All seven key elements, together with their total of 26 dimensions, could be tested and verified through the validation of the case studies in Chapter 5. However, these case studies are limited to the engineering sector and the development of individual software, thus preventing a conclusive determination of whether the identified list of key elements is complete. This statement underscores the dynamic nature of inquiry in this research field and the need for continuous adaption.

Research Question 3: How can the strategic re-engineering approach be adopted into real-world scenarios?

To address this question, the research focused on bridging the gap between theoretical frameworks and practical implementation. The thesis utilized a mixed-methods approach, combining text templates, checklists, models, and prompt questions. These seven practical guidelines were presented consistently at the end of each segment description, whereby the outcomes were summarized in the evaluation template. The usability of the practical guidelines was showcased when validating the methodology in Chapter 5. These guides not only make the methodology more accessible but also drive its adoption as a standard practice in the industry, contributing to digital and sustainable transformation.

6.2 Limitations and Future Research

Particularly because of the numerous existing approaches to redesigning software technology and software engineering in terms of sustainability and its various facets, the need for this methodology arises from the fact that an interplay of humans, technology, and tasks is required. The methodology presented in this thesis should not be seen as a replacement or alternative to existing frameworks or methods. Instead, it is intended to emphasize human beings and their importance, especially in strategic and organizational implementation, and thus create an incentive to think holistically.

While this methodology represents a significant milestone toward achieving a holistic approach that addresses the complex interplay of digitalization and sustainability in software engineering, its limitations must also be acknowledged. The complexity of the twin transformation means that no single methodology can fully address all facets of this challenge. For instance, the methodology may struggle to adapt to standardized software, which could limit its applicability across all scenarios. Future research should focus on refining and expanding this methodology by examining its adaptability to various contexts. In this regard, examples from industries outside the engineering sector are crucial for testing suitability and generalizability. Broadening the field of application can enhance the robustness and reliability of the methodology. Alongside these approaches, there is significant interest in the impact that cutting-edge technologies, particularly generative AI, will have on software technologies in a sustainable context.

Furthermore, in future research projects should consider the quantifiability of the results. Due to the ongoing discussion about the completeness of the key elements, this essential aspect is currently not considered in all methods. As a result, due to the ever-increasing reporting requirements in Europe, robust quantification models should be added to the implementation models.

Keeping in mind these and other possible research projects, the high adaptability of the methodology's architecture should be highlighted once again. The methodology has the option of adding further segments to the defined layers. Such additional segments may emerge during the validation of case studies from a broader field of application that have not emerged or not fully emerged during the specific validation of individual software in mechanical engineering.

The demonstration of the remaining open research topics and the outlook for newly developed research questions underline the dynamic nature of research in this area. This work makes a comprehensible and important contribution on the way to creating a suitable, strategic and, above all, holistic model. Concurrently, despite the feasibility validation through industry-related case studies, it remains open which approach will make the leap into practical application and thus become pioneering for software development in the context of sustainability.

7 Conclusion

The steadily increasing number of research projects in the field of sustainable software indicates that digital and sustainable transformation are becoming increasingly intertwined and are becoming a common challenge in business, science, and society that needs to be mastered. Both transformation trends are characterized by a complex nature with multidimensional influencing factors, which makes integration a challenging task.

After various approaches were presented in Chapter 3, it was established that addressing this task requires a holistic and human-centered approach. The holistic perspective is derived, on the one hand, from the demands of sustainable transformation, which necessitates consideration of the Triple Bottom Line principle encompassing the ecological, economic, and social dimensions. Meanwhile, DT has, through accumulated experience in recent years, increasingly contributed to new insights. It has become evident that successful software systems require not only a technical dimension but must also account for strategic and organizational aspects to be developed and implemented sustainably. These strategic and organizational aspects are primarily attributed to the roles and responsibilities of humans within the methodology.

This approach was pursued to develop a methodology that focuses on awareness during software design, development, and implementation across three layers and seven identified key elements. Consequently, the methodology presented in this thesis is designed for use in software development with a customized software solution intended to be used in the context of a sustainable BM. In this defined environment, it is essential that the software contains sustainable components (sustainability quality attributes and innovation attributes), pursues a business and sustainable value proposition (business and sustainable impact) and is additionally surrounded by enablers who actively contribute to the long-term, targeted, and user-oriented applicability of the software. These enablers are characterized by organizational strategy, resource management and market engagement.

Through the validation of practical case studies, this work has demonstrated that successfully implementing and sustaining the synergy between software and sustainability over the long term requires the software task, software technology, and human enablers to function in harmony.

Bibliography

Abdallah, Y.O., Shehab, E., Al-Ashaab, A., 2022. Sustainable Digital Transformation: The Role of Organisational Digital Culture, in: Advances in Manufacturing Technology XXXV. IOS Press, pp. 60–66.

Abdelkafi, O., Ili, S., Schott, V., 2023. Sustainable Success - The Critical Role of Corporate Carbon Footprint in ROI (Study Report). ILI.DIGITAL AG, Karlsruhe. https://ili.digital/resource/boost-roi-through-corporate-carbon-footprint-innovation/(accessed January 7, 2025).

Abijith, A., Fosso Wamba, S., Gnanzou, D., 2013. A Literature Review on Business Process Management, Business Process Reengineering, and Business Process Innovation. Presented at the Enterprise and Organizational Modeling and Simulation.

Acar, H., 2017. Software development methodology in a Green IT environment. Thesis. Université de Lyon. https://theses.hal.science/tel-01724069/file/TH2017ACARHAYRI.pdf (accessed January 9, 2025).

Adewumi, A., Misra, S., Omoregbe, N., 2015. Evaluating Open Source Software Quality Models Against ISO 25010. IEEE International Conference on Computer and Information Technology. Liverpool, UK, pp. 872-877.

Ajiga, D., Okeleke, P.A., Folorunsho, S. O., Ezeigweneme, C., 2024. Enhancing software development practices with AI insights in high- tech companies. Computer Science & IT Research Journal, 5(8), 1897-1919.

Albertao, F., Xiao, J., Tian, C., Lu, Y., Zhang, K.Q., Liu, C., 2010. Measuring the Sustainability Performance of Software Projects. IEEE 7th International Conference on E-Business Engineering. Shanghai, China, pp. 369–373, doi: 10.1109/ICEBE.2010.26.

Aljarallah, S., 2021. A framework for software sustainability: a case study of e-government and the private sector in KSA. Thesis. Loughborough University. https://repository.lboro.ac.uk/articles/thesis/A_framework_for_software_sustainability_a_case_study_of_e-government_and_the_private_sector_in_KSA/13325348?file=25671665 (accessed January 9, 2025).

Aljarallah, S., Lock, R., 2019. A Comparison of Software Quality Characteristics and Software Sustainability Characteristics, in: Proceedings of the 2019 3rd International Symposium on Computer Science and Intelligent Control. ACM, Amsterdam Netherlands, pp. 1–11, doi: 10.1145/3386164.3389078

Al-Qutaish, R., 2010. Quality Models in Software Engineering Literature: An Analytical and Comparative Study. Journal of American Science 6, pp. 166-175.

Amazon Web Services, Inc, 2024. Front End vs Back End - Difference Between Application Development. https://aws.amazon.com/compare/the-difference-between-frontend-and-backend/ (accessed May 7, 2024).

Amit, R., Zott, C., 2001. Value creation in E-business. Strategic Management Journal 22, pp. 493–520, doi: 10.1002/smj.187.

Amri, R., Bellamine Ben Saoud, N., 2014. Towards a Generic Sustainable Software Model. 4th International Conference on Advances in Computing and Communications 2014, Cochin, India, pp. 231–234, doi: 10.1109/ICACC.2014.62

Ashanin, N., 2019. Quality attributes in Software Architecture. Medium. https://medium.com/@nvashanin/quality-attributes-in-software-architecture-3844ea482732 (accessed July 14, 2024).

Assad, F., Konstantinov, S., Rushforth, E.J., Vera, D.A., Harrison, R., 2021. Virtual engineering in the support of sustainable assembly systems. Procedia CIRP, 8th CIRP Conference of Assembly Technology and Systems 97, pp. 367–372. doi: 10.1016/j.procir.2020.05.252.

Assoratgoon, W., Kantabutra, S., 2023. Toward a sustainability organizational culture model. Journal of Cleaner Production 400, 136666, doi: 10.1016/j.jclepro.2023.136666.

Atadoga, A., Umoga, U.J., Lottu, O.A., Sodiy, E.O., 2024. Tools, techniques, and trends in sustainable software engineering: A critical review of current practices and future directions. World Journal of Advanced Engineering Technology and Sciences 11, pp. 231–239. https://doi.org/10.30574/wjaets.2024.11.1.0051.

Bachmann, N., Tripathi, S., Brunner, M., Jodlbauer, H., 2022. The Contribution of Data-Driven Technologies in Achieving the Sustainable Development Goals. Sustainability 14, 2497, doi: 10.3390/su14052497.

Baumöl, U., Borchers, J., Eicker, S., Hildebrand, K., Jung, R., Lehner, F., 1996. Einordnung und Terminologie des Software Reengineering. Informatik Spektrum 19, pp. 191–195, doi: 10.1007/s002870050030.

Beck, K., Beedle, M., van Bennekum, A., Cunningham, W., Grenning, J., 2001. Manifesto for Agile Software Development. https://agilemanifesto.org (accessed Julyy 14, 2024).

Becker, C., 2014. Sustainability and Longevity: Two Sides of the Same Quality?, Ontario, Canada, doi: 10.13140/2.1.2214.0800.

Becker, C., Betz, S., Chitchyan, R., Duboc, L., Easterbrook, S.M., Penzenstadler, B., Seyff, N., Venters, C.C., 2015. Requirements: The Key to Sustainability. IEEE Software 33, pp.56–65, doi: 10.1109/MS.2015.158.

Belz, C., 2006. Vom Leistungsvorteil zur Value Proposition. Mark Rev St. Gallen 23, pp. 2–6, doi: 10.1007/BF03249114.

Bendor-Samuel, P., 2019. Why digital transformations fail: 3 exhausting reasons | The Enterprisers Project https://enterprisersproject.com/article/2019/8/why-digital-transformations-fail-3-reasons (accessed January 2, 2025).

Bevan, N., 1995. Measuring usability as quality of use. Software Qual J 4, pp. 115–130, doi: 10.1007/BF00402715.

BMZ, 2024. European Green Deal [WWW Document]. Bundesministerium für wirtschaftliche Zusammenarbeit und Entwicklung. https://www.bmz.de/de/service/lexikon/european-green-deal-118284 (accessed October 21, 2024).

Bocken, N., Short, S., Rana, P., Evans, S., 2014. A literature and practice review to develop sustainable business model archetypes. Journal of Cleaner Production 65, pp. 42–56, doi: 10.1016/j.jclepro.2013.11.039.

Boehm, B.W., 1978. Characteristics of Software Quality. North-Holland Publishing Company, Amsterdam, New York, 169 pages.

Böttcher, T., Krcmar, H., 2023. Right Place, Right Time: Configurations of Technology Use for Business Model Innovation. In 29th Annual Americas Conference on Information Systems, AMCIS 2023. Association for Information Systems. https://aisel.aisnet.org/amcis2023/sig_scuidt/sig_scuidt/4/ (accessed December 21, 2024)

Bourque, P., Fairley, R.E., 2014. Guide to the Software Engineering Body of Knowledge - SWEBOK V3.0. IEEE Computer Society Press, 346 pages. https://www.computer.org/education/bodies-of-knowledge/software-engineering (accessed December 21, 2024).

Brunner, M., Bachmann, N., Tripathi, S., Pöchtrager, S., Jodlbauer, H., 2024. Sustainability as a key value proposition - a literature review and potential pathways. Procedia Computer Science, 5th International Conference on Industry 4.0 and Smart Manufacturing (ISM 2023) 232, pp. 1–10, doi: 10.1016/j.procs.2024.01.001.

Bundesumweltministerium, 2020. Rio+20. Bundesministerium für Umwelt, Naturschutz, nukleare Sicherheit und Verbraucherschutz. https://www.bmuv.de/themen/europa-internationales/internationales/rio-20 (accessed December 20, 2022).

Burdea, G.C., Coiffet, P., 2003. Virtual Reality Technology, 2nd Edition. Wiley. https://www.wiley.com/en-us/Virtual+Reality+Technology%2C+2nd+Edition-p-9780471360896 (accessed January 24, 2021).

Calero, C., Bertoa, M.F., Moraga, M.Á., 2013. A systematic literature review for software sustainability measures, in: 2013 2nd International Workshop on Green and Sustainable Software (GREENS). Presented at the 2013 2nd International Workshop on Green and Sustainable Software (GREENS), pp. 46–53, doi: 10.1109/GREENS.2013.6606421.

Chen, A., Li, L., Shahid, W., 2024. Digital transformation as the driving force for sustainable business performance: A moderated mediation model of market-driven business model innovation and digital leadership capabilities. Heliyon 10, e29509, doi: 10.1016/j.heliyon.2024.e29509.

Chesbrough, H., 2010. Business Model Innovation: Opportunities and Barriers, Long Range Planning, 43, pp. 354-363, doi: 10.1016/J.LRP.2009.07.010.

Chesbrough, H., Chesbrough, H., Rosenbloom, R.S., 2002. The Role of the Business Model in Capturing Value from Innovation: Evidence from Xerox Corporation's Technology Spin-Off Companies, Industrial and Corporate Change. 11, doi: 10.1093/ICC/11.3.529.

Chou, Y., 2019. Actionable Gamification: Beyond Points, Badges, and Leaderboards. Createspace Independent Publishing Platform, 2015. https://yukaichou.com/gamification-book/ (accessed March 3, 2025)

Christmann, A.-S., Crome, C., Graf-Drasch, V., Oberländer, A., Schmidt, L., 2024. The Twin Transformation Butterfly. Business & Information Systems Engineering 1–17, doi: 10.1007/s12599-023-00847-2.

Clinton, L., Whisnant, R., 2019. Business Model Innovations for Sustainability, in: Lenssen, G.G., Smith, N.C. (Eds.), Managing Sustainable Business: An Executive Education Case and Textbook. Springer Netherlands, Dordrecht, 463–503, doi: 10.1007/978-94-024-1144-7_22.

Club of Rome, 2024. Earth Overshoot Day 2024. Club of Rome. https://www.clubofrome.org/impact-hubs/climate-emergency/earth-overshoot-day/ (accessed November 3, 2024).

Condori-Fernandez, N., Lago, P., 2019. Towards a Software Sustainability-Quality Model: Insights from a Multi-Case Study, in: 2019 13th International Conference on Research Challenges in Information Science (RCIS), 1–11, doi: 10.1109/RCIS.2019.8877084.

Condori-Fernandez, N., Lago, P., 2018. Characterizing the contribution of quality requirements to software sustainability. Journal of Systems and Software 137, pp. 289–305, doi: 10.1016/j.jss.2017.12.005.

Condori-Fernandez, N., Lago, P., Luaces, M., Catala, A., 2019. A Nichesourcing Framework applied to Software Sustainability Requirements, in: 2019 13th International Conference on Research Challenges in Information Science (RCIS), IEEE, Brussels, Belgium, pp. 1–6, doi: 10.1109/RCIS.2019.8877000.

Dao, V., Langella, I., Carbo, J., 2011. From green to sustainability: Information Technology and an integrated sustainability framework. The Journal of Strategic Information Systems, The Greening of IT 20, pp. 63–79, doi: 10.1016/j.jsis.2011.01.002.

Davila Delgado, J.M., Oyedele, L., Beach, T., Demian, P., 2020. Augmented and Virtual Reality in Construction: Drivers and Limitations for Industry Adoption. Journal of Construction Engineering and Management 146, 04020079, doi: 10.1061/(ASCE)CO.1943-7862.0001844.

Dewett, T., Jones, G.R., 2001. The role of information technology in the organization: a review, model, and assessment. Journal of Management 27, pp. 313–346, doi: 10.1016/S0149-2063(01)00094-0.

Dey, P.K., 1999. Process re-engineering for effective implementation of projects. International Journal of Project Management 17, pp. 147–159, doi: 10.1016/S0263-7863(98)00023-4.

Die Bundesregierung, 2022. Rückblick auf bisherige Klimakonferenzen | Bundesregierung. Die Bundesregierung informiert | Startseite. https://www.bundesregierung.de/breg-de/themen/klimaschutz/klimakonferenzen-rueckblick-1965144 (accessed December 30, 2022).

Duboc, L., Penzenstadler, B., Porras, J., Akinli Kocak, S., Betz, S., Chitchyan, R., Leifler, O., Seyff, N., Venters, C., 2020. Requirements engineering for sustainability: an awareness framework for designing software systems for a better tomorrow. Requirements Engineering 25, pp. 469–492, doi: 10.1007/s00766-020-00336-y.

Dutonde, P.D., 2022. Website Developmemt Technologies: A Review. International Journal for Research in Applied Science & Engineering Technology (IJRASET) 10, pp. 359–366, doi:10.22214/ijraset.2022.39839.

Dyllick, T., Hockerts, K., 2002. Beyond the Business Case for Corporate Sustainability. Business Strategy and the Environment, 11, pp. 130–141, doi: 10.1002/bse.323.

Ebert, C., 2014. Systematisches Requirements Engineering: Anforderungen ermitteln, spezifizieren, analysieren und verwalten, 5th ed. dpunkt-Verlag. Heidelberg. 467 pages. ISBN 3864901391 | 9783864901393.

Echterhoff, B., 2018. Methodik zur Einführung innovativer Geschäftsmodelle in etablierten Unternehmen Vol 387. Verlagsschriftenreihe des Heinz Nixdorf Instituts, Paderborn. https://digital.ub.uni-paderborn.de/hs/content/titleinfo/2996916 (accessed October 2, 2024).

Elkington, J., 2018. 25 Years Ago I Coined the Phrase "Triple Bottom Line." Here's Why It's Time to Rethink It. Harvard Business Review. https://hbr.org/2018/06/25-years-ago-i-coined-the-phrase-triple-bottom-line-heres-why-im-giving-up-on-it (accessed September 5, 2024).

Elkington, J., 1999. Cannibals with Forks: The Triple Bottom Line of 21st Century Business. Capstone. 424 pages. https://www.wiley.com/en-ie/Cannibals+with+Forks%3A+The+Triple+Bottom+Line+of+21st+Century+Business-p-9781841120843 (accessed October 10, 2024).

El-Namaki, M.S.S., 1992. Creating a corporate vision. Long Range Planning 25, 25–29, doi: 10.1016/0024-6301(92)90166-Y.

Elser, S., 2022. Frontend vs. Backend vs. Full-Stack - Der Unterschied erklärt. ElevateX. https://elevatex.de/de/blog/it/frontend-vs-backend-vs-fullstack-unterschiede/ (accessed January 3, 2025).

Evans, S., Vladimirova, D., Holgado, M., Van Fossen, K., Yang, M., Silva, E.A., Barlow, C.Y., 2017. Business Model Innovation for Sustainability: Towards a Unified Perspective for Creation of Sustainable Business Models. Business Strategy and the Environment 26, pp. 597–608, doi: 10.1002/bse.1939.

Garcia, J., 2022. Common pitfalls in transformations: A conversation with Jon Garcia | McKinsey. https://www.mckinsey.com/capabilities/transformation/our-insights/common-pitfalls-in-transformations-a-conversation-with-jon-garcia (accessed January 2, 2025).

Gassmann, O., Frankenberger, K., Csik, M., 2017. Geschäftsmodelle entwickeln: 55 innovative Konzepte mit dem St. Galler Business Model Navigator. pp. 1–14, doi: 10.3139/9783446452848.fm.

Gerstlacher, J., Groher, I., Plösch, R., 2022. Green und Sustainable Software im Kontext von Software Qualitätsmodellen. HMD 59, pp. 1149–1164, doi: 10.1365/s40702-021-00821-0.

Global Footprint Network, 2024. Press Release Earth Overshoot Day 2024. Earth Overshoot Day. https://overshoot.footprintnetwork.org/newsroom/press-release-2024-english/ (accessed January 2, 2025).

Goodwin, N.C., 1987. Functionality and usability. Commun. ACM 30, pp. 229–233, doi: 10.1145/214748.214758.

Grand View Research, 2024. Digital Transformation Market Size And Share Report, 2030. https://www.grandviewresearch.com/industry-analysis/digital-transformation-market (accessed January 2, 2025).

Grothus, A., Thesing, T., Feldmann, C., 2021. Digitale Geschäftsmodell-Innovation mit Augmented Reality und Virtual Reality: Erfolgreich für die Industrie entwickeln und umsetzen. Springer Berlin Heidelberg, Berlin, Heidelberg, doi: 10.1007/978-3-662-63746-3.

Häfner, P., 2021. Holistic Approach for Authoring Immersive and Smart Environments for the Integration in Engineering Education, Karlsruher Institut für Technologie (KIT), 352 pages, doi: 10.5445/IR/1000130947. https://publikationen.bibliothek.kit.edu/1000130947 (accessed September 7, 2024).

Hagiu, A., Wright, J., 2023. To Get Better Customer Data, Build Feedback Loops into Your Products. https://hbr.org/2023/07/to-get-better-customer-data-build-feedback-loops-into-your-products (accessed September 7, 2024).

Hammer, M., Champy, J., 1993. Reengineering the corporation: A manifesto for business revolution. Business Horizons 36, pp. 90–91, doi: 10.1016/S0007-6813(05)80064-3.

He, B., Zheng, H., Tang, K., Xi, P., Li, M., Wei, L., Guan, Q., 2024. A Comprehensive Review of Lithium-Ion Battery (LiB) Recycling Technologies and Industrial Market Trend Insights. Recycling 9, 9, doi: 10.3390/recycling9010009.

Heany, D.F., 1983. Degrees of product innovation. Journal of Business Strategy 3, pp. 3–14, doi: 10.1108/eb038984.

Hess, T., Matt, C., Benlian, A., Wiesböck, F., 2016. Options for Formulating a Digital Transformation Strategy. MIS Quarterly Executive 15, pp. 123–139.

Hochwarth, D., 2024. Mercedes-Benz startet CO_2-neutrales Recycling von E-Auto-Batterien. https://www.ingenieur.de/technik/fachbereiche/verkehr/mercedes-benz-startet-co%E2%82%82-neutrales-recycling-von-e-auto-batterien/ (accessed January 10, 2025).

Horner, H., 2024. Power Ahead with These 2024 Mechanical Engineering Trends! Engineering Institute of Technology. https://www.eit.edu.au/power-ahead-with-mechanical-engineering-trends/ (accessed July 28, 2024).

Howey, R., 2002. Understanding software technology. Knowledge, Technology & Policy 15, pp. 70–81, doi: 10.1007/s12130-002-1006-0.
Hui, K.L., Tam, K.Y., 2002. Software Functionality: A Game Theoretic Analysis. Journal of Management Information Systems 19, pp. 151–184.

Humphrey, W.S., 1988. The software engineering process: definition and scope. SIGSOFT Softw. Eng. Notes 14, pp. 82–83, doi: 10.1145/75111.75122.

Ili, S., 2009. Open Innovation im Kontext der Integrierten Produktentwicklung: Strategien zur Steigerung der FuE-Produktivität, Karlsruher Institut für Technologie (KIT), doi: 10.5445/IR/1000013586. https://publikationen.bibliothek.kit.edu/1000013586 (accessed May 6, 2024).

Imran, A., Kosar, T., 2019. Software Sustainability: A Systematic Literature Review and Comprehensive Analysis, doi: 10.48550/arXiv.1910.06109.

International Organization for Standardization, 2023. ISO/IEC 25010:2023. ISO. https://www.iso.org/standard/78176.html (accessed May 7, 2024).

Isensee, C., Teuteberg, F., Griese, K.-M., Topi, C., 2020. The relationship between organizational culture, sustainability, and digitalization in SMEs: A systematic review. Journal of Cleaner Production 275, 122944, doi: 10.1016/j.jclepro.2020.122944.

Jayal, A.D., Badurdeen, F., Dillon, O.W., Jawahir, I.S., 2010. Sustainable manufacturing: Modeling and optimization challenges at the product, process and system levels. CIRP Journal of Manufacturing Science and Technology, Sustainable Development of Manufacturing Systems 2, pp. 144–152, doi: 10.1016/j.cirpj.2010.03.006.

Khalifeh, A., 2020. Advances in business management: The incorporation of sustainability in software engineering projects and the potential impact on project success in the context of Jordanian public Universities. Ph.D. Thesis, University of Bolton, Bolton, UK.

https://core.ac.uk/reader/350904808 (accessed October 5, 2024).

Khalifeh, A., Al-Adwan, A.S., Alrousan, M.K., Yaseen, H., Mathani, B., Wahsheh, F.R., 2023. Exploring the Nexus of Sustainability and Project Success: A Proposed Framework for the Software Sector. Sustainability 15, 15957, doi: 10.3390/su152215957.

Khalifeh, A., Farrell, P., Alrousan, M., Alwardat, S., Faisal, M., 2020. Incorporating sustainability into software projects: a conceptual framework. International Journal of Managing Projects in Business 13, pp. 1339-1361, doi: 10.1108/IJMPB-12-2019-0289.

Kocak, S.A., Alptekin, G.I., Bener, A.B., 2015. Integrating Environmental Sustainability in Software Product Quality. Fourth International Workshop on Requirements Engineering for Sustainable Systems (RE4SuSy) 1216. Ottawa, Canada.

Kolasani, S., 2023. Leadership in business innovation and transformation, navigating complex digital landscapes and enterprise technology ecosystems and achieving sustainable growth in today’s rapidly evolving market. International Journal of Holistic Management Perspectives 4, pp. 1–23. https://injmr.com/index.php/IJHMP/article/view/57 (accessed January 7, 2025).

Koziolek, H., 2011. Sustainability evaluation of software architectures: a systematic review, in: Proceedings of the Joint ACM SIGSOFT Conference, Boulder Colorado USA, pp. 3–12, doi: 10.1145/2000259.2000263.

Krodel, T., Schott, V., Mayer, A., Ovtcharova, J., 2024. Impact of XR-Enabled Collaboration in Businesses—An Economic, Ecological, and Social Perspective, in: Alareeni, B., Elgedawy, I. (Eds.), AI and Business, and Innovation Research: Understanding the Potential and Risks of AI for Modern Enterprises. Springer Nature Switzerland, Cham, pp. 767–777, doi: 10.1007/978-3-031-42085-6_66.

Lammert, D., 2024. Bridging Academic Software Sustainability Design With Corporate Business Planning, Ph.D. Thesis, Furtwangen University, doi:10.13140/RG.2.2.33322.63680. https://www.researchgate.net/publication/379665617_Bridging_Academic_Software_Sustainability_Design_With_Corporate_Business_Planning (accessed January 7, 2025).

Lammert, D., Betz, S., Porras, J., 2024a. The Business-Oriented Extension of the Sustainability Awareness Framework—A Design Science Study. pp. 47–64, doi: 10.1007/978-981-99-7886-1_5.

Lammert, D., Betz, S., Porras, J., Oyedeji, S., 2024b. Sustainability in the Software Industry: A Survey Study on the Perception, Responsibility, and Motivation of Software Practitioners. Engineering Intelligent Systems 32, pp. 77-84.

Laukkanen, M., Tura, N., 2022. Sustainable value propositions and customer perceived value: Clothing library case. Journal of Cleaner Production 378, 134321, doi: 10.1016/j.jclepro.2022.134321.

Laukkanen, M., Tura, N., 2020. The potential of sharing economy business models for sustainable value creation. Journal of Cleaner Production 253, 120004, doi: 10.1016/j.jclepro.2020.120004.

Lee, J., Ceyhan, P., Jordan-Cooley, W., Sung, W., 2013. GREENIFY A Real-World Action Game for Climate Change Education. Simulation & Gaming 44, pp. 349–365, doi: 10.1177/1046878112470539.

Lee, M.-C., 2014. Software Quality Factors and Software Quality Metrics to Enhance Software Quality Assurance. BJAST 4, pp. 3069–3095, doi: 10.9734/BJAST/2014/10548.

Leong, J., May Yee, K., Baitsegi, O., Palanisamy, L., Ramasamy, R.K., 2023. Hybrid Project Management between Traditional Software Development Lifecycle and Agile Based Product Development for Future Sustainability. Sustainability 15, 1121, doi: 10.3390/su15021121.

Liu, Y., 2022. The Impact of Digital Leadership on Enterprise Sustainable Development Under the Background of “Industry 4.0.” 6th International Seminar on Education, Management and Social Sciences (ISEMSS 2022), Atlantis Press, pp. 1445–1454, doi: 10.2991/978-2-494069-31-

2_171.

Lu, T., Gupta, A., Jayal, A.D., Badurdeen, F., Feng, S.C., Jr., O.W.D., Jawahir, I.S., 2011. A Framework of Product and Process Metrics for Sustainable Manufacturing, in: Seliger, G., Khraisheh, M.M.K., Jawahir, I.S. (Eds.), Advances in Sustainable Manufacturing. Springer, Berlin, Heidelberg, pp. 333–338, doi: 10.1007/978-3-642-20183-7_48.

Lüdeke-Freund, F., 2010. Towards a Conceptual Framework of Business Models for Sustainability ERSCP-EMSU conference, Delft, The Netherlands, October 25-29, doi: 10.13140/RG.2.1.2565.0324.

Lyytinen, K., Yoo, Y., Boland, R., 2015. Digital product innovation within four classes of innovation networks. Information Systems Journal 26, doi: 10.1111/isj.12093.

Majuntke, V., Obermaier, F., 2023. Towards Sustainable Software. Proceedings Series of the Gesellschaft für Informatik. Bonn. doi: 10.18420/inf2023_15.

MarketsandMarkets, 2024. Digital Transformation Market. https://www.marketsandmarkets.com/Market-Reports/digital-transformation-market-43010479.html (accessed January 2, 2025).

Mayer, J.H., 2012. Using the Kano Model to Identify Attractive User-Interface Software Components. In J. F. George (Ed.), Proceedings of the 33rd International Conference on Information Systems (ICIS 2012). https://aisel.aisnet.org/icis2012/proceedings/HumanComputerInteractions/2/ (accessed January 4, 2025).

McCall, J.A., Richards, P.K., Walters, G.F., 1977. Factors in Software Quality. Volume I. Concepts and Definitions of Software Quality. General Electric CO Sunnyvale CA. https://apps.dtic.mil/sti/citations/ADA049014 (accessed October 5, 2024).

McCartney, A., 2023. Die 10 wichtigsten strategischen Technologie-Trends von Gartner für 2024. Gartner. https://www.gartner.de/de/artikel/die-10-wichtigsten-strategischen-technologie-trends-von-gartner-fuer-2024 (accessed July 28, 2024).

McGuire, S., Schultz, E., Ayoola, B., Ralph, P., 2023. Sustainability is Stratified: Toward a Better Theory of Sustainable Software Engineering. Presented at the 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE), IEEE Computer Society, 1996–2008, doi: 10.1109/ICSE48619.2023.00169.
Miehe, R., Waltersmann, L., Sauer, A., Bauernhansl, T., 2021. Sustainable production and the role of digital twins–Basic reflections and perspectives. Journal of Advanced Manufacturing and Processing 3, e10078, doi: 10.1002/amp2.10078.

Mohamed, K.S., 2020. New Frontiers in Cryptography: Quantum, Blockchain, Lightweight, Chaotic and DNA. Springer International Publishing, Cham, doi: 10.1007/978-3-030-58996-7.

Moises De Souza, A.C., Soares Cruzes, D., Jaccheri, L., Krogstie, J., 2024. Social Sustainability Approaches for Software Development: A Systematic Literature Review, in: Kadgien, R., Jedlitschka, A., Janes, A., Lenarduzzi, V., Li, X. (Eds.), Product-Focused Software Process Improvement, Lecture Notes in Computer Science. Springer Nature Switzerland, Cham, 478–494, doi: 10.1007/978-3-031-49266-2_33.

Moreira, A., Araújo, J., Gralha, C., Goulão, M., Brito, I.S., Albuquerque, D., 2023. A social and technical sustainability requirements catalogue. Data & Knowledge Engineering 143, 102107, doi: 10.1016/j.datak.2022.102107.

Morgenstern, E., 2007. The origin and early application of the principle of sustainable forest management. The Forestry Chronicle 83, pp. 485–489, doi: 10.5558/tfc83485-4.

Motta, R.C., De Oliveira, K.M., Travassos, G.H., 2019. A conceptual perspective on interoperability in context-aware software systems. Information and Software Technology 114, pp. 231–257, doi: 10.1016/j.infsof.2019.07.001.

Naumann, S., Dick, M., Kern, E., Johann, T., 2011. The GREENSOFT Model: A reference model for green and sustainable software and its engineering. Sustainable Computing: Informatics and Systems 1, pp. 294–304, doi: 10.1016/j.suscom.2011.06.004.

Nishant, R., Kennedy, M., Corbett, J., 2020. Artificial intelligence for sustainability: Challenges, opportunities, and a research agenda. International Journal of Information Management 53, 102104, doi: 10.1016/j.ijinfomgt.2020.102104.
Noman, H., Mahoto, N., Bhatti, S., Rajab, A., Shaikh, A., 2024. Towards sustainable software systems: A software sustainability analysis framework. Information and Software Technology 169, 107411, doi: 10.1016/j.infsof.2024.107411.

Nylén, D., Holmström, J., 2014. Digital innovation strategy: A framework for diagnosing and improving digital product and service innovation. Business Horizons 58. doi: 10.1016/j.bushor.2014.09.001.

Osterwalder, A., 2004. The Business Model Ontology – A Proposition in a Design Science Approach. PhD Thesis, University of Lausanne, Switzerland, 286 pages. https://d1wqtxts1xzle7.cloudfront.net/30373644/thebusiness-model-ontology-libre.pdf?1391809579=&response-content-disposition=inline%3B+filename%3DThe_Business_Model_Ontology_a_propositio.pdf&Expires=1751990163&Signature=WyDRViP-bUU2lj0ShDtSjSSGVx2OGYaTLPHchyR7sQ5UIIqlKfJPNg6bPpSuCO-Ziw3WwsefY~-fdq2UBeg~plBycIM2y48NWkHXMhStagGPFjTht3RJpYmulbAOpYK-JPFs4Bgp2O1zDO0Dhfh-bBwP-gaFizCv~NDPu~y4YxAM3iNYp0xiJrkF2mUW2YuZ2xCTAC97BnxG4wH4eCM3u2ydHPeZOZJrMg33mdUAb7tdHzraTSYCBjr-ZIZwMTXVlO6AOYoobMF~X2Xkd6psXLPJOnuIHlh-HWWdr-BACvZKeqsV7rMAg2GESGviM-fIgtCgQf8qhUhHwr~xO~Go9IeQv4z4Q__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA (accessed March 3, 2024).

Osterwalder, A., Pigneur, Y., 2010. Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers. Wiley. 288 pages. https://www.wiley.com/en-us/Business+Model+Generation%3A+A+Handbook+for+Visionaries%2C+Game+Changers%2C+and+Challengers-p-9780470876411 (accessed March 3, 2024).

Osterwalder, A., Pigneur, Y., Bernarda, G., Smith, A., 2015. Value Proposition Design: How to Create Products and Services Customers Want. John Wiley & Sons. https://www.strategyzer.com/library/value-proposition-design-2 (accessed March 3, 2024).

Ousterhout, J.K., 2018. A philosophy of software design, First edition. ed. Yaknyam Press, Palo Alto, CA. https://milkov.tech/assets/psd.pdf (accessed January 8, 2025).

Oxford English Dictionary, 2024. sustainable, adj. meanings, etymology and more | Oxford English Dictionary. https://www.oed.com/dictionary/sustainable_adj?tl=true (accessed January 8, 2025).

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

Paneru, Biplov, Paneru, Bishwash, Poudyal, R., Shah, K.B., 2024. Exploring the Nexus of User Interface (UI) and User Experience (UX) in the Context of Emerging Trends and Customer Experience, Human Computer Interaction, Applications of Artificial Intelligence. International Journal of Informatics, Information System and Computer Engineering (INJIISCOM) 5, 102–113, doi: 10.34010/injiiscom.v5i1.12488.

Patala, S., Jalkala, A., Keränen, J., Väisänen, S., Tuominen, V., Soukka, R., 2016. Sustainable value propositions: Framework and implications for technology suppliers. Industrial Marketing Management 59, doi: 10.1016/j.indmarman.2016.03.001.

Paul, D., Tan, Y.L., 2015. An Investigation of the Role of Business Analysts in IS Development. European Conference on Information Systems, doi: 10.18151/7217442.

Paulk, M.C., 1994. A Comparison of ISO 9001 and the Capability Maturity Model for Software. Software Engineering Institute. Carnegie Mellon University Pittsburgh, Pennsylvania. https://insights.sei.cmu.edu/documents/1111/1994_005_001_435267.pdf (accessed January 9, 2025).

Penzenstadler, B., 2013. Towards a Definition of Sustainability in and for Software Engineering, In Proceedings of the 28th Annual ACM Symposium on Applied Computing (SAC '13). Association for Computing Machinery, New York, NY, USA, pp. 1183–1185, doi: 10.1145/2480362.2480585.

Penzenstadler, B., Bauer, V., Calero, C., Franch, X., 2012. Sustainability in software engineering: a systematic literature review, in: 16th International Conference on Evaluation & Assessment in Software Engineering (EASE 2012), Ciudad Real, Spain, pp. 32–41, doi: 10.1049/ic.2012.0004.

Penzenstadler, B., Femmer, H., 2013. A Generic Model for Sustainability with Process- and Product-specific Instances, In Proceedings of the 2013 workshop on Green in/by software engineering (GIBSE '13). Association for Computing Machinery, New York, NY, USA, pp. 3–8, doi: 10.1145/2451605.2451609.

Petrov, I., Samoilenko, V., Kotelnikova, I., Tomakh, V., Bocharova, N., 2023. Leadership's Role in Navigating Sustainability and Digitalisation in Enterprise. International Journal of Organizational Leadership 12, pp. 165–182, doi: 10.33844/ijol.2023.60378.

Piepponen, A., Ritala, P., Keränen, J., Maijanen, P., 2022. Digital transformation of the value proposition: A single case study in the media industry. Journal of Business Research 150, pp. 311–325, doi: 10.1016/j.jbusres.2022.05.017.

Pothukuchi, A.S., Kota, L.V., Mallikarjunaradhya, V., 2023. Impact of Generative AI on the Software Development Lifecycle (SDLC). International Journal of Creative Research Thoughts 11.

Potting, J., Hekkert, M.P., Worrell, E., Hanemaaijer, A., 2017. Circular Economy: Measuring Innovation in the Product Chain. Planbureau voor de Leefomgeving, 2544.

Ramesh, N., Delen, D., 2021. Digital Transformation: How to Beat the 90% Failure Rate? IEEE Engineering Management Review 49, 22–25, doi: 10.1109/EMR.2021.3070139.

Reinitz, B., 2020. Consider the Three Ds When Talking about Digital Transformation. EDUCAUSE Review. https://er.educause.edu/blogs/2020/6/consider-the-three-ds-when-talking-about-digital-transformation (accessed January 7, 2025).

Robinson, H., 2019. Why do most transformations fail? A conversation with Harry Robinson | McKinsey. https://www.mckinsey.com/capabilities/transformation/our-insights/why-do-most-transformations-fail-a-conversation-with-harry-robinson (accessed January 7, 2025).

Rosário, A.T., Dias, J.C., 2022. Sustainability and the Digital Transition: A Literature Review. Sustainability 14, 4072, doi: 10.3390/su14074072.

Schaltegger, S., Lüdeke-Freund, F., Hansen, E., 2012. Business Cases for Sustainability: The Role of Business Model Innovation for Corporate Sustainability. International Journal of Innovation and Sustainable Development 6, pp. 95–119, doi: 10.1504/IJISD.2012.046944.

Schiuma, G., Santarsiero, F., Carlucci, D., Jarrar, Y., 2024. Transformative leadership competencies for organizational digital transformation. Business Horizons 67, pp. 425–437, doi: 10.1016/j.bushor.2024.04.004.

Schott, V., Ovtcharova, J., 2024. Integrating Sustainability into Software Development: A Global Categorization, in: Mansour, N., Bujosa Vadell, L.M. (Eds.), Finance and Law in the Metaverse World: Regulation and Financial Innovation in the Virtual World. Springer Nature Switzerland, Cham, pp. 59–69, doi: 10.1007/978-3-031-67547-8_6.

Seaborn, K., Fels, D., 2015. Gamification in Theory and Action: A Survey. International Journal of Human-Computer Studies 74, pp. 14–31, doi: 10.1016/j.ijhcs.2014.09.006.

Sharma, W., Lim, W.M., Kumar, S., Verma, A., Kumra, R., 2024. Game on!

A state-of-the-art overview of doing business with gamification. Technological Forecasting and Social Change 198, 122988, doi: 10.1016/j.techfore.2023.122988.

Siemens, R.C., 2024. Digitalisierung der CPG-Fertigung. Siemens Resource Center. https://resources.sw.siemens.com/de-DE/video-consumer-products-retail-digitalization-cpg-manufacturing (accessed July 21, 2024).

Silveira, C., Santos, V., Reis, L., Mamede, H., 2022. CRESustain: Approach to Include Sustainability and Creativity in Requirements Engineering. J. Engg. Res. & Sci. 1, pp. 27–34, doi: 10.55708/js0108004.

Silvi, R., Nielsen, C., Pia, A., 2023. From Insights to Business Model Innovation and Results: Using the Digital Transformation Canvas. Journal of Business Models 11, pp. 30–45.

Sneed, H., Verhoef, C., 2001. Reengineering the Corporation - A Manifesto for IT Evolution. https://www.researchgate.net/publication/2366189_Reengineering_the_Corporation_-_A_Manifesto_for_IT_Evolution (accessed January 3, 2025).

Sneed, H.M., 1991. Economics of software re-engineering. Journal of Software Maintenance: Research and Practice 3, pp. 163–182, doi: 10.1002/smr.4360030304.

Soegaard, M., 2024. The Role of Micro-interactions in Modern UX. The Interaction Design Foundation. https://www.interaction-design.org/literature/article/micro-interactions-ux (accessed August 10, 2024).

Software Sustainability Institute, 2024. About the Software Sustainability Institute. https://www.software.ac.uk/about (accessed October 7, 2024).

Somani, A.K., Vaidya, N.H., 1997. Understanding Fault Tolerance and Reliability. Computer 30, pp. 45–50, doi: 10.1109/MC.1997.585153.

Sommerville, I., 2016. Software engineering, 10th Edition, Pearson Education Limited, Boston, 811 pages. https://dn790001.ca.archive.org/0/items/bme-vik-konyvek/Software%20Engineering%20-%20Ian%20Sommerville.pdf (accessed January 3, 2025).

Soongpol, B., Netinant, P., Rukhiran, M., 2024. Practical Sustainable Software Development in Architectural Flexibility for Energy Efficiency Using the Extended Agile Framework. Sustainability 16, 5738, doi: 10.3390/su16135738.

Souza, A., Koochakpour, K., Papavlasopoulou, S., Jaccheri, L., 2024. Exploring Social Sustainability Alignment in Software Development Projects, pp. 250–261, doi: 10.5220/0012739600003687.

Sriraman, G., Raghunathan, S., 2023. A Systems Thinking Approach to Improve Sustainability in Software Engineering—A Grounded Capability Maturity Framework. Sustainability 15, 8766,
doi: 10.3390/su15118766.

Steininger, D., 2018. Linking Information Systems and Entrepreneurship: A Review and Agenda for IT-Associated and Digital Entrepreneurship Research. Information Systems Journal 29, pp. 363–407,
doi: 10.1111/isj.12206.

Sundar, S.S., Marathe, S.S., 2010. Personalization versus Customization: The Importance of Agency, Privacy, and Power Usage. Human Communication Research 36, pp. 298–322,
doi: 10.1111/j.1468-2958.2010.01377.x.

Teece, D., 2010. Business Models, Business Strategy and Innovation. Long Range Planning 43, pp. 172–194, doi: 10.1016/j.lrp.2009.07.003.

Tenzer, F., 2024. Daten - Volumen der weltweit generierten Daten bis 2027. Statista. https://de.statista.com/statistik/daten/studie/267974/umfrage/prognose-zum-weltweit-generierten-datenvolumen/ (accessed October 8, 2024).

Timmers, P., 1998. Business Models for Electronic Markets. Elec. Markets 8, pp. 3–8, doi: 10.1080/10196789800000016.

Toniolo, K., Masiero, E., Massaro, M., Bagnoli, C., 2020. Sustainable Business Models and Artificial Intelligence: Opportunities and Challenges, in: Knowledge, People, and Digital Transformation. Approaches for a Sustainable Future., Contributions to Management Science. Springer, Cham, pp. 103–117, doi: 10.1007/978-3-030-40390-4_8.

United Nations, 2015. THE 17 GOALS | Sustainable Development [WWW Document]. URL https://sdgs.un.org/goals (accessed January 3, 2025).

United Nations, 1987. Report of the World Commission on Environment and Development. https://digitallibrary.un.org/record/139811?ln=en&v=pdf (accessed January 5, 2025).

Venters, C., Lau, L., Griffiths, M., Holmes, V., Ward, R., Jay, C., Dibsdale, C., Xu, J., 2014. The Blind Men and the Elephant: Towards an Empirical Evaluation Framework for Software Sustainability. Journal of Open Research Software 2, doi: 10.5334/jors.ao.

Venters, C.C., Capilla, R., Nakagawa, E.Y., Betz, S., Penzenstadler, B., Crick, T., Brooks, I., 2023. Sustainable software engineering: Reflections on advances in research and practice. Information and Software Technology 164, 107316, doi: 10.1016/j.infsof.2023.107316.

Vial, G., 2019. Understanding digital transformation: A review and a research agenda. The Journal of Strategic Information Systems 28, pp. 118–144, doi: 10.1016/j.jsis.2019.01.003.

Vladimirova, D., 2019. Building Sustainable Value Propositions for Multiple Stakeholders: A Practical Tool. Journal of Business Models 7, pp. 1–8, doi: 10.5278/ojs.jbm.v7i1.2103.

Wang, P., 2019. On Defining Artificial Intelligence. Journal of Artificial General Intelligence 10, pp. 1–37, doi: 10.2478/jagi-2019-0002.

Wang, Y., 2024. Operating Systems, in: The Engineering Handbook. Routledge & CRC Press, p. 15.

Weill, P., Woerner, S., 2013. Optimizing Your Digital Business Model. MIT Sloan Management Review 54, pp. 28–35, doi: 10.1109/EMR.2015.7059380.

Welch, J., Rayo, M.F., Kanter, B., Bagian, T., 2016. Framework for Alarm Management Process Maturity. Biomedical Instrumentation & Technology 165–179, doi: 10.2345/0899-8205-50.3.165.

Wohlin, C., Šmite, D., Moe, N.B., 2015. A general theory of software engineering: Balancing human, social and organizational capitals. Journal of Systems and Software 109, pp. 229–242, doi: 10.1016/j.jss.2015.08.009.

Yassien, E., 2020. The challenges of capability maturity model integration application in the dynamic environment. International Journal of Information Systems and Change Management 12, pp. 17–34, doi: 10.1504/IJISCM.2020.112045.

Zhang, X., Badurdeen, F., Rouch, K., Jawahir, I.S., 2013. On improving the product sustainability of metallic automotive components by using the total life-cycle approach and the 6R methodology. Presented at the 11th Global Conference on Sustainable Manufacturing, Berlin, p. 6.

Zott, C., Amit, R., Massa, L., 2011. The Business Model: Recent Developments and Future Research. Journal of Management, 37, 1019-1042, doi: 10.1177/0149206311406265.