Vol. 8, No. 4, August 2026

ISI Journal of Information Systems and Informatics

Published By Asosiasi Doktor Sistem Informasi Indonesia

Green Software Engineering for Sustainable IT Services: A Scopus-Based Systematic Literature Review and Bibliometric Mapping

Cecep Muhamad Sidik Ramdani[1], I Made Oka Widyantara[2], Linawati[3], Nyoman Putra Sastra[4]

[1]Department of Data Science, Siliwangi University, Tasikmalaya, Indonesia

[2,3,4]Department of Electrical Engineering, Udayana University, Denpasar, Indonesia

Received:
February 17, 2026
Revised:
May 27, 2026
Accepted:
July 22, 2026
Published:
August 22, 2026

Corresponding Author:
Author Name*:
I Made Oka Widyantara
Email*:
oka.widyantara@unud.ac.id

DOI:
10.63158/journalisi.v8i4.1672

Abstract. This study maps the development of Green Software Engineering (GSE) research and examines its contribution to sustainable IT services. A Scopus-based Systematic Literature Review (SLR) was combined with bibliometric mapping. Records were retrieved on May 6, 2026 using TITLE-ABS-KEY("Green Software"), screened according to PRISMA, and analyzed with VOSviewer. After applying document-type and language criteria, 108 English-language Scopus-indexed journal articles were included. The findings indicate that GSE is an emerging but steadily growing field, with publication activity increasing markedly after 2016. Spain was the leading contributor, while major research themes included energy efficiency, energy consumption, sustainable development, software design, Green Computing, software sustainability, and Green IT. This study contributes by integrating systematic review evidence with bibliometric mapping to provide a consolidated and traceable overview of GSE in the context of sustainable IT services. Practically, the findings highlight research-informed priorities for IT service planning, including energy efficiency, hardware efficiency, carbon awareness, sustainability measurement, and climate commitments. The results are limited to Scopus-indexed evidence retrieved using the keyword "Green Software" and should not be interpreted as representing all global GSE research.

Keywords: Green Software Engineering; Sustainable IT Services; Energy Efficiency; Systematic Literature Review; Bibliometric Analysis.

a) INTRODUCTION

Green Software, Green Software Engineering, and sustainable software are related but not identical concepts. Green Software refers to software whose development, operation, and use seek to reduce environmental impact through lower energy consumption, efficient resource use, and carbon-aware behavior [1], [2], [3]. Green Software Engineering is the systematic application of engineering principles, methods, metrics, and tools to design, implement, measure, and maintain such environmentally responsible software across the Software Development Life Cycle (SDLC). Sustainable software has a broader scope because it also covers social, economic, technical, and individual sustainability dimensions in addition to environmental concerns. In the context of sustainable IT services, Green Software Engineering directly supports service sustainability by reducing energy demand in applications and supporting infrastructure, improving software-driven hardware utilization, enabling measurement of service-related carbon impacts, and guiding organizations toward more efficient digital operations.

Research on Green Software has expanded across software engineering, Green IT, sustainable computing, artificial intelligence, Internet of Things (IoT), edge computing, and digital transformation. However, the existing literature is still dispersed across technical themes such as coding practices, design patterns, code smells, refactoring, software architecture, energy measurement, and organizational adoption barriers [4], [5], [6], [7], [8], [9], [10], [11], [12], [13]. This dispersion makes it difficult to see how Green Software Engineering can be translated into sustainable IT service practices. Therefore, a focused synthesis is needed to clarify the development of the field, identify underexplored issues, and connect software-level sustainability evidence with IT service planning.

The main gap addressed in this study is the limited availability of Scopus-based reviews that integrate qualitative SLR synthesis with bibliometric mapping for Green Software Engineering in the context of sustainable IT services. Previous studies have provided important technical and conceptual insights, but they have not sufficiently combined publication trends, country and institutional contributions, source distribution, author productivity, keyword co-occurrence, and synthesized Green Software attributes within one traceable SLR-bibliometric map. This integrated approach adds value because it combines interpretive evidence synthesis with measurable research-pattern analysis

while keeping the scope limited to Scopus-indexed publications retrieved using the keyword "Green Software."

Accordingly, this study examines Green Software Engineering research using three operational research questions: RQ1: How has Green Software Engineering research developed over time, and does the publication trend indicate continuing significance for future scholarly inquiry? RQ2: How are Scopus-indexed Green Software publications distributed across countries, institutional affiliations, publication sources, authors, and keyword networks? RQ3: What theoretical and practical implications can be derived from the SLR and bibliometric findings for future Green Software Engineering research and sustainable IT services? RQ1 is answered through publication trend analysis, RQ2 through bibliometric mapping of distribution and networks, and RQ3 through synthesis of dominant themes, gaps, and Green Software attributes.

To answer these questions, this study applies a Scopus-based SLR combined with bibliometric analysis. The SLR is used to organize evidence, identify conceptual and practical gaps, and synthesize implications from the reviewed literature. Bibliometric analysis is required as a complementary method because it makes the structure of the field visible through measurable indicators such as publication growth, country contribution, author productivity, source distribution, and keyword co-occurrence. The contribution of this study is twofold: theoretically, it consolidates the dispersed Green Software Engineering literature into an integrated map of themes and research directions; practically, it offers evidence-based guidance for organizations seeking to strengthen sustainable IT services through energy-efficient, hardware-efficient, measurable, carbon-aware, and climate-aligned software practices.

b) METHODS

2.1. Literature Review

The “Digital Green” framework was introduced as a quantitative approach for assessing the environmental consequences of digitalizing business processes, particularly in the context of business process reengineering. When applied to a case study at a small university in southern Italy, the framework demonstrated its ability to estimate whether digital transformation produces environmental benefits or drawbacks. Its carbon-aware

orientation is based on the use of energy consumption as a proxy for carbon emissions, thereby linking digital operations to ecological impact. However, the study acknowledges limited generalizability because the framework was tested within a specific business process and industry context [14]. Another study developed a model for estimating the energy consumption of Model-View-Controller (MVC) applications without executing them. The MVC-CCsEM model showed that software size, source lines of code (SLOC), cyclomatic complexity, and maintainability are relevant predictors of energy consumption, while automated metric calculation using tools such as SonarQube was emphasized as important for improving measurement accuracy [15].

The configuration that produces the highest speedup does not necessarily result in the lowest energy consumption, indicating a trade-off between performance and energy efficiency. Substantial energy savings can be achieved under optimal parallel configurations, which highlights the importance of energy-aware optimization strategies in software execution [16]. Behavior-based consumption profiles (BBCPs) have also been proposed to model software behavior using time-based, stochastic, and adaptive constraints. This approach supports early energy-consumption estimation without depending on a specific hardware or software platform, making it useful for software architects and designers who aim to integrate sustainability from the initial design stage [17].

GreSDN was introduced as an energy-saving scheme for Software Defined Networks by applying the Constant Weight Greedy Algorithm and the Improved Constant Weight Greedy Algorithm. The improved algorithm achieved 16% to 36% higher energy savings than Dijkstra's algorithm and approached optimal performance in simulation results [18]. Another study revealed that although most software practitioners consider sustainability important, many are still unable to identify sustainability requirements and often confuse Green Software with sustainable software. To address this issue, a catalog was proposed to support the identification of sustainability requirements [19]. Green Software also remains underused in software development education, although execution time and memory usage can serve as indicators of programming competence. Data mining techniques can also help predict student performance and potentially reduce data center use in massive online learning environments [20].

Foundational elements of the Green Software process have been identified through a pilot study, resulting in five main factors: resources, people, organization, technical elements, and environment. These factors can serve as benchmarks for evaluating sustainable software processes [1]. Word-based text compression has also been shown to substantially reduce energy consumption during search operations, with compressed text requiring less energy than plain text [21]. In addition, an SNMP-based monitoring framework has demonstrated extensibility for network management [22], while the Batch Operations energy design pattern has been shown to reduce the carbon footprint of mobile communication peripherals [23]. Collectively, these studies indicate that Green Software research covers various areas, including modeling, coding, architecture, networking, education, measurement, and carbon-aware design.

Table 1. Paraphrased Defining Elements of Green Software

Defining factor of Green Software	Reference
Green Software refers to software development and use that reduces environmental burden through energy efficiency, system sustainability, and the integration of sustainability into digitalization.	[23]
Green Software is designed to lower energy consumption and environmental impact by using sustainability metrics and practices that support energy efficiency and zero-emission ambitions in ICT.	[15]
Green Software promotes sustainable design through architecture-aware strategies that balance performance optimization with reduced power demand and environmental impact.	[16]
Green Software development integrates green design principles from the early SDLC stages to produce software that is both sustainable and energy efficient.	[17]
Some studies discuss energy-saving mechanisms in SDN rather than defining Green Software directly, showing the breadth of related but indirect contributions.	[18]
Green Software is often treated as part of sustainable software, but sustainability extends beyond environmental concerns to include social, economic, technical, and individual dimensions.	[19]

Defining factor of Green Software	Reference
Green Software involves efficient development and use with minimal environmental impact while encouraging better programming practices.	[20]
Green Software emphasizes software life-cycle practices that reduce environmental impact, improve resource efficiency, and integrate economic, social, and environmental concerns.	[1]
Green Software supports energy-efficient development that minimizes consumption while preserving functionality, particularly in data processing and retrieval.	[21]
Green Software can also support efficient network management and monitoring by optimizing resource use in environmentally friendly software systems.	[22]
Green Software Engineering develops guidelines and good practices for reducing energy use, managing carbon impact, and minimizing the environmental footprint of computing systems.	[24]
Green Software Engineering improves sustainability by measuring energy and hardware efficiency, optimizing architecture, and reducing the impact of energy-intensive applications.	[25]
Green Software reduces waste and energy consumption during both development and operation, supporting sustainability in processes and computing environments.	[10]
Green Software can improve hardware energy efficiency by embedding green practices in software engineering and database design.	[26]
Green Software is a sustainability-oriented development approach that uses green metrics and energy-efficient practices to reduce ecological impact.	[27]

2.2. Research Design

This study applied a Scopus-based Systematic Literature Review (SLR) supported by bibliometric mapping to identify the development, distribution, thematic structure, and implications of Green Software Engineering research for sustainable IT services. The formal review followed a PRISMA-oriented workflow to support transparent

identification, screening, eligibility checking, and inclusion. The data were retrieved from Scopus on May 6, 2026, using the focused query TITLE-ABS-KEY("Green Software"), which searched the article title, abstract, and keyword fields. The broader Green IT search shown in Figure 1 was used only as preliminary domain exploration and was not treated as the formal inclusion set for the final Green Software corpus. Scopus was selected because it provides structured bibliographic metadata suitable for SLR documentation and bibliometric analysis, including author names, affiliations, countries, sources, citations, abstracts, author keywords, and index keywords. This workflow follows the PRISMA 2020 reporting framework [28].

The use of the single keyword "Green Software" was deliberate. Related terms such as sustainable software engineering, green coding, energy-aware software, carbon-aware software, and sustainable software may broaden the corpus but can also retrieve studies that discuss general sustainability, energy systems, hardware, or organizational sustainability without explicitly positioning the work within Green Software research. Therefore, this study used "Green Software" as a focused and traceable entry point to map the core Scopus-indexed literature that explicitly adopts this term. This decision strengthens conceptual consistency but is also recognized as a methodological limitation because some relevant studies using alternative terminology may not be captured.

The inclusion criteria were: (1) publications indexed in Scopus and retrieved on May 6, 2026; (2) English-language documents; (3) documents explicitly addressing Green Software in the title, abstract, or keywords; and (4) documents categorized as journal articles. The exclusion criteria were: (1) documents not containing the specific keyword "Green Software" in the selected Scopus fields; (2) non-article document types, including conference papers, review articles, book chapters, books, proceedings, conference reviews, editorials, notes, and short surveys; (3) non-English documents; (4) duplicate records identified through DOI, EID, title, and author matching; and (5) records that did not provide sufficient bibliographic metadata for bibliometric processing. Although only one database was used, potential duplicates were still checked after export by comparing DOI, EID, title, and author information. No duplicate records were retained in the final corpus; when duplicate-like metadata appeared, the most complete Scopus record was prioritized.

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

Eligibility checking for the SLR component was conducted as relevance screening, not as a full methodological quality assessment of empirical study designs. Each record was assessed based on its alignment with the study scope, namely Green Software, Green Software Engineering, software energy efficiency, software-related carbon awareness, software measurement, and sustainable IT service relevance. Records were considered eligible when their title, abstract, keywords, or available bibliographic information indicated a direct relationship with Green Software as a software-oriented sustainability concept. Records were excluded when the keyword appeared only incidentally or when the document type, language, or metadata did not meet the eligibility requirements. This relevance-screening process strengthened the conceptual fit of the corpus, while the bibliometric component was used to map publication patterns and keyword relationships.

Bibliometric analysis was conducted using VOSviewer to generate network visualizations and quantify relationships among bibliographic elements. The analysis used full counting rather than fractional counting, meaning that each occurrence of an author, country, source, institution, or keyword was counted as one full contribution. The normalization method used in the visualizations was association strength normalization. For readability, the following thresholds were applied to the visualized maps: a minimum of five occurrences for keyword co-occurrence, four documents for country analysis, four articles for author productivity, four articles for institutional affiliation mapping, and four articles for source analysis. Items below these thresholds remained in the dataset but were excluded from the visual maps to reduce label density. For keyword co-occurrence, both author keywords and Scopus index keywords were considered, and terms with equivalent meaning, spelling variation, singular-plural forms, or capitalization differences were harmonized before mapping. General bibliometric procedures were aligned with established guidance [29], and VOSviewer was used as the software reference for constructing and visualizing bibliometric maps [30].

The five Green Software attributes discussed in this study, namely energy efficiency, hardware efficiency, measurement, climate commitments, and carbon awareness, were derived through a hybrid deductive-inductive synthesis. Deductively, the initial attribute categories were informed by Green Software Foundation concepts and recurring sustainability themes in Green Software Engineering. Inductively, these categories were refined by reviewing the included literature and identifying repeated concepts, metrics,

challenges, and implications across the corpus. Therefore, the attributes should not be interpreted solely as direct quotations from one framework; rather, they represent a synthesized interpretation of Scopus-indexed Green Software literature supported by established Green Software concepts.

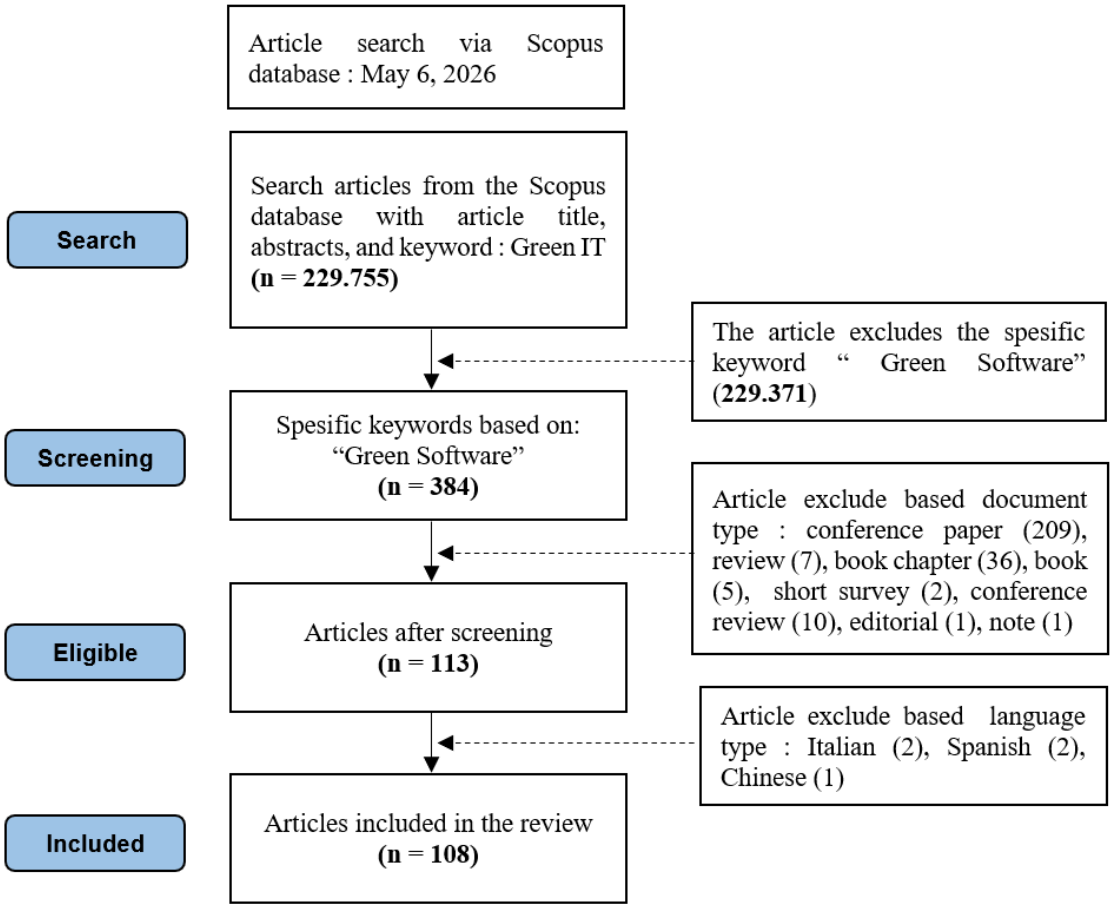

Figure 1. Systematic Literature Review Information Flow Using PRISMA

Figure 1 presents the PRISMA-oriented information flow used in this study. The initial 229,755 Green IT records represent preliminary domain exploration and should be distinguished from the formal Green Software search. The formal screening began with Scopus records containing the specific keyword "Green Software" in the title, abstract, and keyword fields (n = 384). During eligibility assessment, 271 records were excluded because they were not journal articles, including conference papers (209), review articles (7), book chapters (36), books (5), short surveys (2), conference reviews (10), editorials (1), and notes (1), leaving 113 article records. A further language-based screening excluded

five non-English articles, consisting of Italian (2), Spanish (2), and Chinese (1) documents. Consequently, 108 English-language Scopus-indexed journal articles were included in the final review and bibliometric mapping. The possible 2006 SuperState record was retained only for transparent bibliometric accounting because it appeared in the Scopus keyword-based retrieval, but it was treated as a false-positive item in the substantive SLR synthesis and was not used to derive the Green Software attributes.

c) RESULTS AND DISCUSSION

3.1. Bibliometric Mapping

This section reports the results of the Scopus-based SLR and bibliometric mapping of 108 English-language journal articles on Green Software. To maintain terminological consistency, the term articles is used for the final included corpus, while records or documents refer to items identified during the earlier PRISMA screening stages. The analysis covers descriptive statistics, annual publication trends, country and institutional contributions, source distribution, author productivity, keyword co-occurrence, cluster interpretation, source productivity, and the SLR-derived Green Software attributes relevant to sustainable IT services.

Table 2. Descriptive Statistics of the Scopus-Indexed Green Software Corpus

Indicator	Result	Explanation
Total final articles	108	English-language Scopus-indexed journal articles included after PRISMA screening.
Year range	2006-2026	Publication years covered by the Scopus export retrieved on May 6, 2026.
Document type	Journal articles only	Conference papers, review articles, book chapters, books, proceedings, editorials, notes, short surveys, and non-English records were excluded.
Database	Scopus	The corpus was obtained using TITLE-ABS-KEY("Green Software").

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

Indicator	Result	Explanation
Total authors	257	Unique author count identified from the Scopus export used in the bibliometric analysis.
Total citations	2164	Total citation count recorded in the Scopus export at the time of data retrieval.
Average citations per article	20.04	Average citations per article calculated as 2,164 total citations divided by 108 articles.

3.2. Research Question

1) RQ1: How has Green Software Engineering research developed over time, and does the publication trend indicate continuing significance for future scholarly inquiry?

The publication trend indicates that Green Software Engineering remains an emerging but growing research area. The final corpus comprises 108 journal articles published between 2006 and May 6, 2026. Because the article entitled "SuperState: A computer program for the control of operant behavioral experimentation" appeared in the Scopus search results but does not substantively discuss Green Software Engineering, it is treated as a possible keyword-based false-positive [31]. It was retained only as part of the transparent bibliometric screening history but was not used as substantive evidence for deriving Green Software attributes. The more meaningful development of the field is reflected in studies after 2016, when attention increasingly shifted toward software energy efficiency, software-related carbon awareness, measurement, hardware efficiency, and sustainable IT services.

Table 3 complements Figure 2 by making the annual publication trend explicit. The distribution confirms that Green Software research was sparse during the early years and became more visible after 2016. The peak in 2025 suggests increasing scholarly attention, while the lower number in 2026 should be interpreted cautiously because the dataset was retrieved in May 2026 and does not represent a complete publication year.

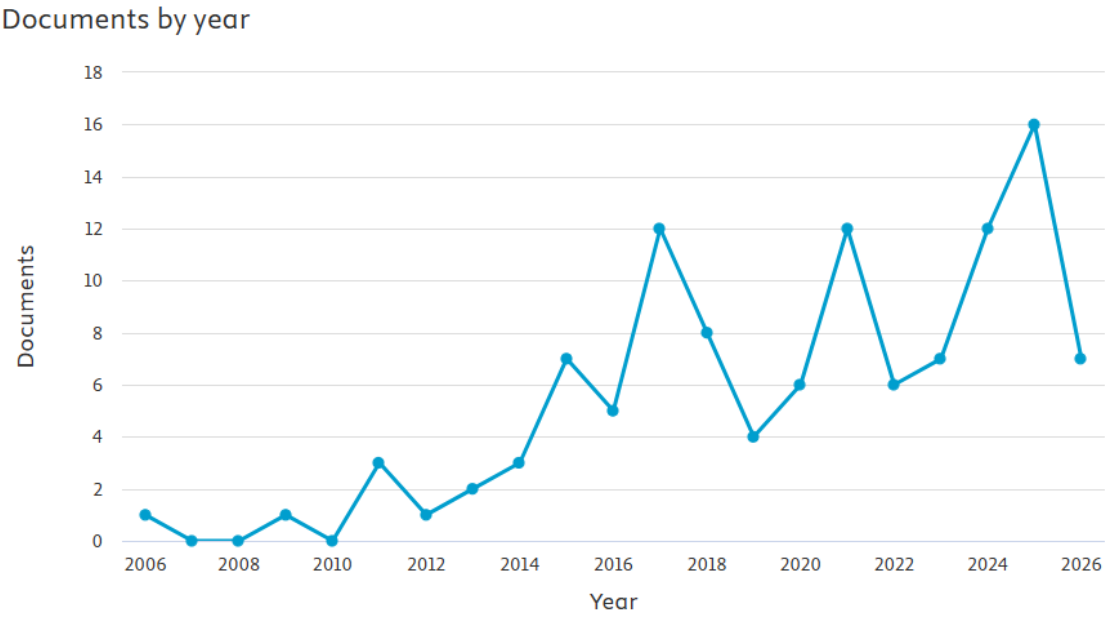

Figure 2. Number of Green Software Publications

Table 3. Annual Publication Trend of Green Software Articles

Year	Articles	Year	Articles	Year	Articles
2006	1	2013	1	2020	6
2007	0	2014	3	2021	12
2008	0	2015	6	2022	6
2009	1	2016	4	2023	7
2010	0	2017	12	2024	11
2011	2	2018	8	2025	16
2012	1	2019	4	2026*	7
Total				108	

2) RQ2: How are Scopus-indexed Green Software publications distributed across countries, institutional affiliations, publication sources, authors, and keyword networks?

The distribution of the 108 English-language Scopus-indexed journal articles was analyzed across countries, institutional affiliations, publication sources, productive authors, and keyword co-occurrence networks. In this article, country contribution is based on the countries recorded in the Scopus affiliation metadata and counted in VOSviewer using full counting. Therefore, a publication with authors from multiple countries may contribute one full count to each represented country; the results do not rely only on the

corresponding author's country. This clarification is important because the reported country distribution reflects all author affiliations captured in Scopus rather than a single-author or corresponding-author allocation.

Using VOSviewer full counting of Scopus affiliation countries, Spain appears as the most dominant contributor with 37 articles, followed by China with 12 articles, India with 11 articles, Malaysia with 9 articles, Germany with 7 articles, Pakistan with 6 articles, Italy, the Netherlands, and Switzerland with 5 articles each, and Australia with 4 articles, as presented in Figure 3. Because full counting was applied, these values should be interpreted as country participation counts in the Scopus-indexed corpus, not as exclusive article ownership by a single country.

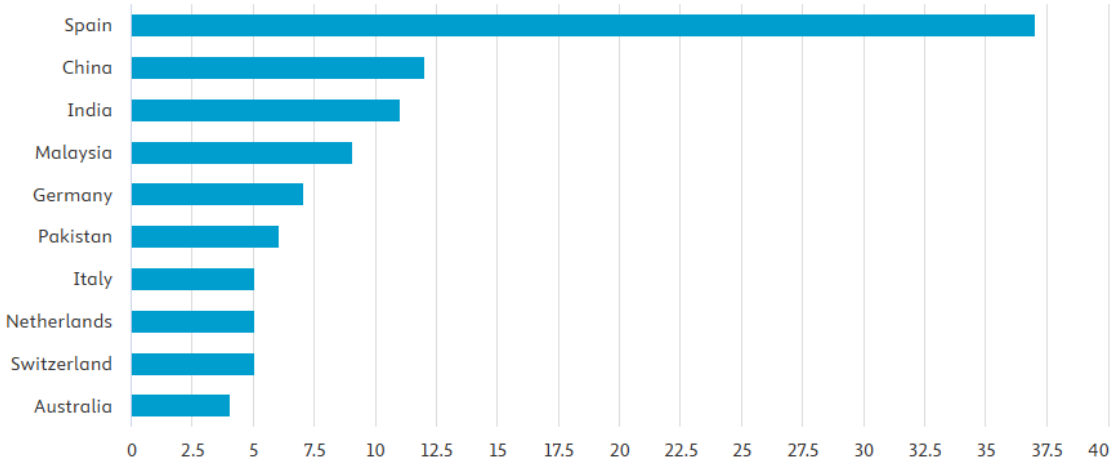

Figure 3. Number of Green Software Publications by Country

The country-level distribution shows that Green Software research is led by Spain but has also gained attention in countries with different technological capacities. Contributions from China, India, Malaysia, Germany, Pakistan, Italy, the Netherlands, Switzerland, and Australia demonstrate the global relevance of this topic. The VOSviewer country network further indicates collaboration and connectivity among countries involved in Green Software research. This mapping is important for designing future international and comparative research agendas. Figure 4 further shows that international collaboration is uneven. Spain is positioned as a central node in the country network, indicating strong connectivity with other contributors, while several countries appear at the periphery with fewer collaborative links. This pattern suggests that Green

Software research has developed through a limited number of strong country clusters rather than through a fully distributed global collaboration network.

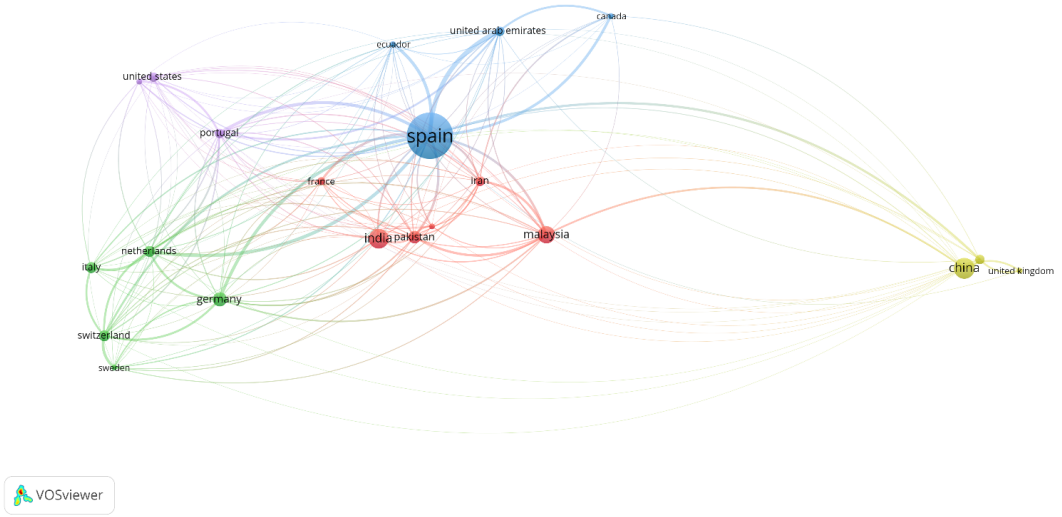

Figure 4. Network Country Visualization

These findings suggest that Green Software principles are not confined to highly industrialized or technologically advanced countries. They are also gaining attention in developing technology contexts, indicating that Green Software can be adapted across diverse national settings and used to support more inclusive software models.

Institutional affiliation analysis, also based on Scopus affiliation metadata and full counting, shows that Universidad de Castilla-La Mancha (Spain) leads with 15 articles, followed by Universiti Kebangsaan Malaysia and Universidad de Cádiz with 6 articles each. Other active institutions include Universidad de Murcia, Hochschule Trier, Universiti Malaysia Terengganu, University of Malakand, Vrije Universiteit Amsterdam, Universität Zürich, and Universidade do Minho. The prominence of Spanish institutions is consistent with the dominant role of Spain in the country-level analysis.

The affiliation distribution indicates that Green Software is studied by institutions in both advanced and developing technology contexts. However, the concentration of outputs in a relatively small group of institutions also shows that the field is not yet evenly distributed. This creates an opportunity for broader empirical validation of Green Software practices in underrepresented regions and organizational settings.

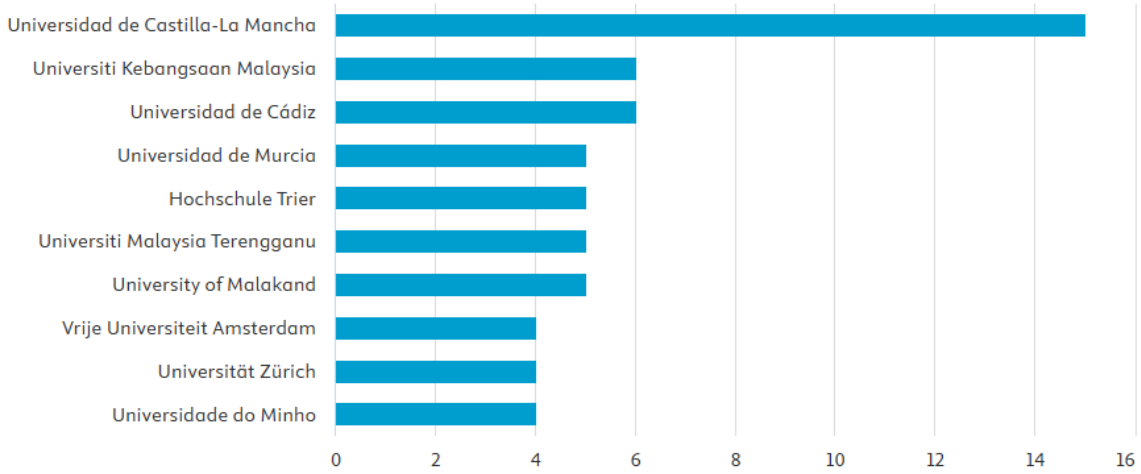

Figure 5. Number of Publications by Affiliation

Source productivity analysis shows that Green Software articles are concentrated in a small number of publication outlets. Sustainable Computing: Informatics and Systems, Journal of Systems and Software, and Information and Software Technology each contain 6 articles, followed by IEEE Access with 5 articles and Applied Sciences Switzerland with 4 articles, as shown in Figure 6. This source analysis supports the bibliometric mapping claim by identifying the publication venues that most frequently disseminate Green Software research, although source productivity should not be interpreted as journal impact factor or citation impact.

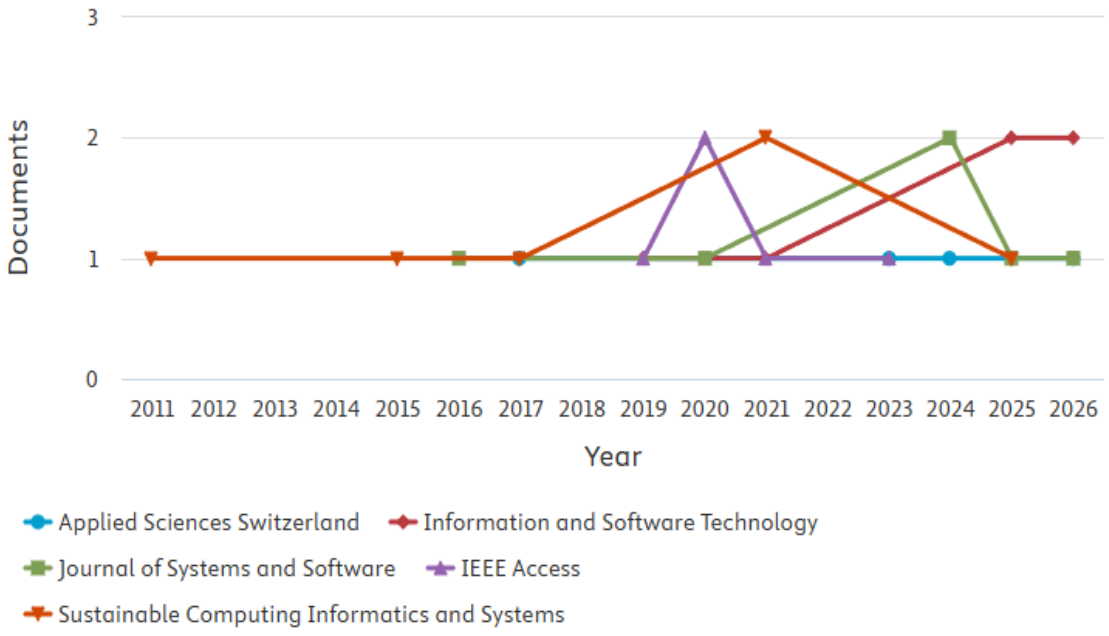

Figure 6. Number of Articles by Sources

Author productivity and co-authorship patterns indicate that the field is shaped by several recurring scholars and research groups. Coral Calero is the most productive author with 15 articles, followed by Garcia with 9 articles, Naumann and Yahaya with 6 articles each, and Deraman, Dorronsoro, Kern, Khan, Mancebo, and Dick with 5 articles each, as displayed in Figure 7. The dominance of these authors suggests that Green Software research has developed around identifiable expert communities rather than a very broad and mature author base.

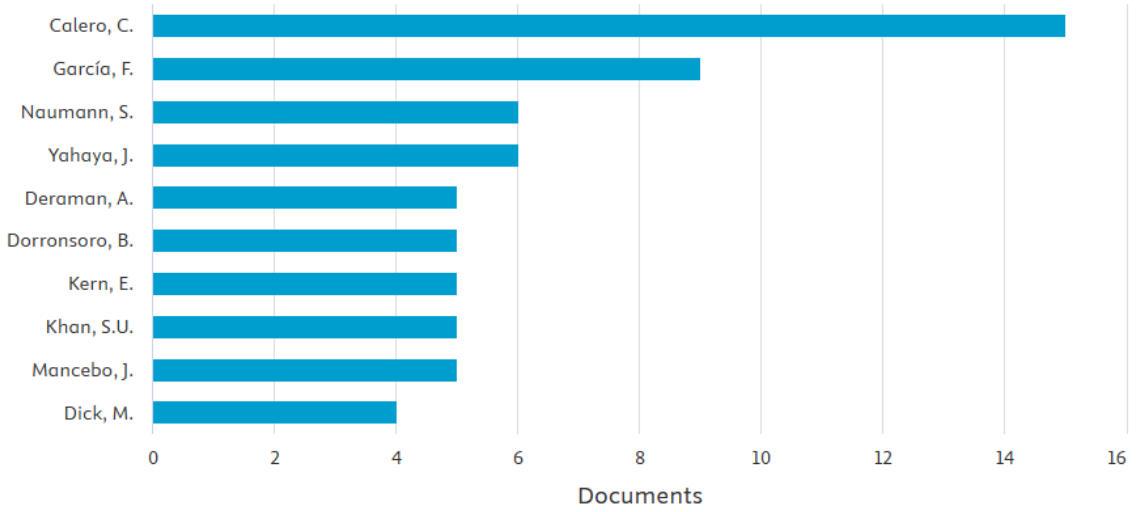

Figure 7. Count of Publications by Author (Top 10 Authors)

3) RQ3: What theoretical and practical implications can be derived from the SLR and bibliometric findings for future Green Software Engineering research and sustainable IT services?

The analysis of 108 Scopus-indexed journal articles using VOSviewer indicates that Green Software research has theoretical and practical implications for sustainable IT services. Theoretically, the bibliometric results reveal the dominant intellectual structure of the field, while the SLR synthesis clarifies how recurring concepts can be translated into Green Software attributes. Practically, the findings help organizations identify which software-related sustainability priorities are most visible in the literature, including energy efficiency, hardware efficiency, measurement, climate commitments, and carbon awareness.

Figure 8 and Table 4 should be interpreted as keyword co-occurrence and total link strength results, not as simple keyword frequency counts. In VOSviewer, total link

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

strength indicates the overall strength of connections between a keyword and other keywords in the network. Thus, a keyword with high total link strength is not merely frequent; it is also strongly connected to other thematic terms in the corpus. Energy Efficiency (432), Green Software (403), Energy Utilization (335), Sustainable Development (253), Software Design (205), Energy Consumption (197), Green Computing (189), Software Engineering (128), Software Sustainability (90), and Green IT (81) represent the most strongly connected terms in the co-occurrence network.

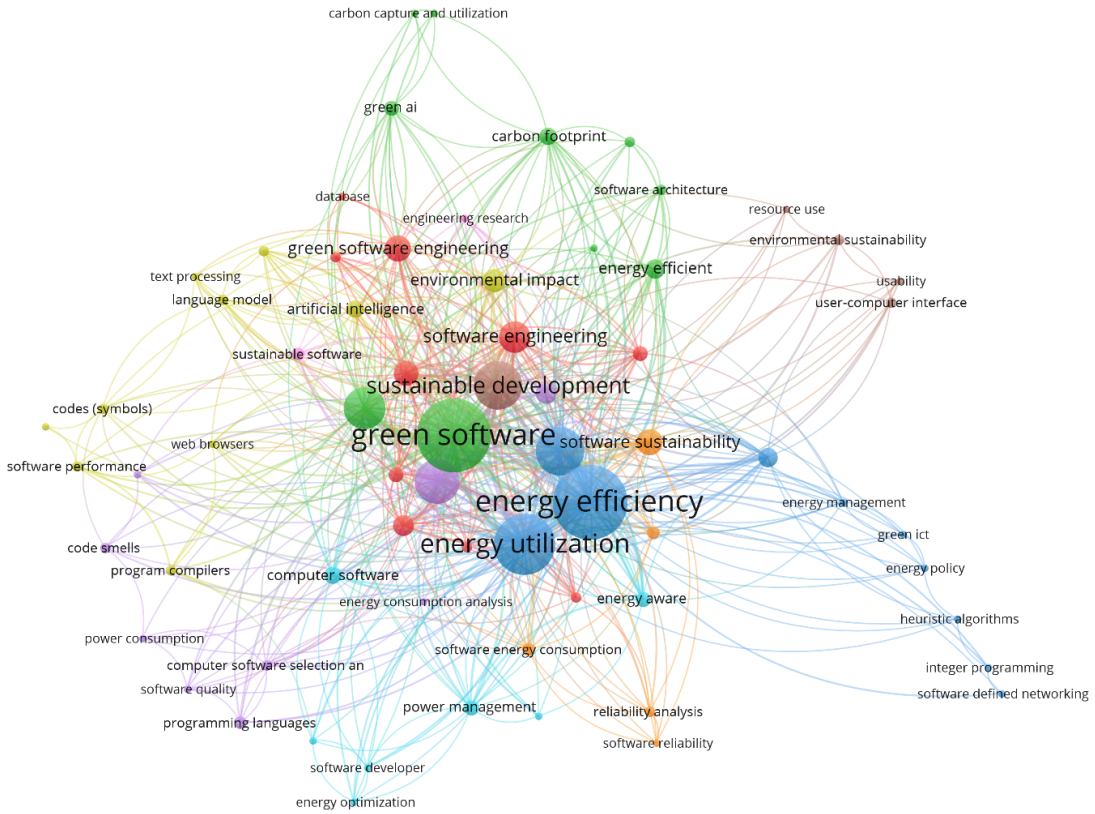

Figure 8. Network Co-occurrence Framework and Representation of Key Terms

Table 4. Top Keywords by Total Link Strength in the VOSviewer Co-occurrence Network

Rank	Keyword	Total Link Strength
1	Energy Efficiency	432
2	Green Software	403
3	Energy Utilization	335
4	Sustainable Development	253
5	Software Design	205
6	Energy Consumption	197

Rank	Keyword	Total Link Strength
7	Green Computing	189
8	Software Engineering	128
9	Software Sustainability	90
10	Green IT	81

The co-occurrence map in Figure 8 can be interpreted through several thematic clusters rather than through a list of top keywords only. The clusters show that Green Software research is organized around energy optimization, software design and engineering, sustainable computing and Green IT, measurement and evaluation, and carbon- or climate-oriented software. These clusters also reveal an important gap: many previous studies remain concentrated in European institutions and technical energy-efficiency topics, while organizational adoption, Green DevOps, sustainable cloud services, AI energy consumption, and empirical validation in developing regions remain less visible.

Table 5. Interpretation of VOSviewer Keyword Co-occurrence Clusters

Cluster	Dominant keywords	Interpretation	Research implications
Energy optimization	energy efficiency; energy utilization; energy consumption	Focuses on reducing energy use during software execution and service operation.	Link energy optimization with service-level indicators.
Software design and engineering	software design; software engineering; software sustainability	Connects Green Software with architecture, coding, and SDLC practices.	Validate green design and coding practices empirically.
Sustainable computing and Green IT	sustainable development; Green Computing; Green IT	Relates software sustainability to broader Green IT concerns.	Integrate software metrics with IT service management.
Measurement and evaluation	measurement; metrics; performance; hardware efficiency	Emphasizes the need to verify energy, resource,	Develop standardized and repeatable

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

Cluster	Dominant keywords	Interpretation	Research implications
		and emission impacts.	measurement models.
Carbon and climate orientation	carbon awareness; carbon footprint; climate commitments	Highlights carbon-aware and climate-aligned software development.	Validate carbon-aware practices in real organizations.

The five attributes shown in Figure 9 were derived from the SLR evidence through a hybrid deductive-inductive synthesis. Deductively, the initial categories were informed by Green Software Foundation concepts and recurring Green Software Engineering principles. Inductively, the included articles were reviewed to identify repeated concepts, metrics, challenges, and implementation implications. Energy efficiency emerged from studies emphasizing software energy use and energy optimization; hardware efficiency emerged from studies linking software behavior with hardware utilization; measurement emerged from work on metrics, tools, and evaluation processes; climate commitments emerged from literature connecting software practice with sustainability and emission-reduction goals; and carbon awareness emerged from studies addressing carbon impact, carbon-aware software, and responsible digital service operation. Therefore, these attributes represent a synthesized interpretation of the SLR corpus rather than a direct copy of a single framework.

Figure 9. Green Software Attributes

a) Energy Efficiency

Energy efficiency refers to using less energy to achieve the same or better output. It is a central strategy for addressing global energy problems, mitigating climate change, and supporting sustainable development. In practice, energy efficiency includes reducing energy loss, optimizing consumption, lowering demand, and adopting technologies such as recycling, waste heat recovery, smart grids, artificial intelligence, and IoT-based energy management systems. It is widely applied in buildings, industry, and transportation to reduce energy demand, carbon emissions, and operating costs. Its benefits are environmental, economic, and social because it can lower greenhouse gas emissions, reduce infrastructure needs, improve competitiveness, strengthen energy security, and increase affordability. However, implementation may be hindered by high initial costs, limited training, organizational resistance, and unequal distribution of costs.

b) Hardware Efficiency

Hardware efficiency describes the ability of computing systems to deliver high performance while minimizing resource use, especially energy consumption. This issue is increasingly critical because high-performance computing, data centers, mobile systems, and heterogeneous architectures require substantial energy. Techniques such as parallel computing, dynamic voltage and frequency scaling, hardware accelerators, approximate computing, heterogeneous task allocation, and dynamic partial reconfiguration in FPGA systems can improve efficiency. Hardware efficiency is closely connected to software because algorithms, programming languages, and architecture-specific optimization influence energy use and performance. Compiled languages and optimized algorithms may provide better energy savings, while heterogeneous systems can allocate workloads to the most efficient processors. Nevertheless, challenges remain due to the slowing of Moore's Law, the end of Dennard scaling, complex HPC architectures, and the need for accurate measurement tools.

c) Measurement

Green Software measurement refers to assessing software greenness through energy efficiency, resource use, and environmental sustainability. Reliable measurement is essential because the environmental benefits of Green Software cannot be confirmed without systematic assessment mechanisms. Existing frameworks and tools include green metrics for energy efficiency, resource efficiency, cost reduction, usability, productivity,

employee support, and tool support, as well as tools such as Green-JEXJ, GreenMiner, and GreenDash for measuring software energy use and design sustainability. The Green Software Measurement Process (GSMP) provides structured procedures to ensure repeatable and comparable measurements. Learning-based approaches can also analyze green metrics during runtime and recommend improvements. However, standardization gaps and complex environments such as virtualization and containers continue to challenge accurate measurement.

d) Climate Commitments

Climate commitments in Green Software refer to development principles and practices aimed at reducing energy use, lowering carbon emissions, and contributing to climate mitigation goals. Green Software Engineering therefore focuses on designing, developing, and maintaining software that reduces life-cycle environmental impacts through energy-efficient design, optimized resources, green coding, and sustainable process management. These commitments matter because software indirectly shapes ICT emissions through hardware use, energy demand, and carbon output. Climate-oriented Green Software requires sustainability to be embedded in quality control, architecture, and development processes, including reducing waste, complexity, inefficient algorithms, and excessive resource use. Challenges include weak standardization, limited awareness, insufficient training, economic barriers, legacy systems, and the lack of accepted emission-measurement frameworks. Emerging initiatives such as the Green Software Foundation, EcoDocSense, and the Shared Responsibility of Software Emissions framework help guide emission reduction and clarify stakeholder responsibility.

e) Carbon Awareness

Carbon awareness in Green Software refers to understanding, measuring, and reducing carbon emissions associated with software across development, deployment, operation, and maintenance. Software affects emissions directly through runtime energy consumption and indirectly through hardware use, infrastructure demand, data processing, and SDLC activities. Carbon-aware software development encourages green coding, computational optimization, reduced unnecessary resource use, and sustainability integration in design and quality requirements. Tools such as GreenDash, energy feedback systems, and software-based energy monitoring can help evaluate the sustainability of components such as databases, hardware, networks, and user interactions. However,

adoption is limited by weak industry-wide implementation, non-standard metrics, insufficient educational resources, and limited integration into current software engineering practices. The growing use of AI and code generation further increases the need to consider energy use and sustainability metrics in complex software systems.

3.3. Discussion

The findings indicate that Green Software Engineering is still an emerging but strategically important field for sustainable IT services. The bibliometric pattern shows a movement from general concerns about Green IT and sustainable computing toward more software-specific concerns, including energy efficiency, software design, measurement, hardware efficiency, and carbon awareness. This pattern suggests that the theoretical development of Green Software Engineering is still being consolidated: the field is no longer limited to broad environmental claims, but is gradually moving toward measurable software engineering concepts that can be linked to design decisions, runtime behavior, infrastructure utilization, and service sustainability.

The dominance of Spain in the corpus should be interpreted critically. Based on the country and institutional distribution, Spain appears as a leading contributor because several productive institutions and recurring authors are strongly represented in the Scopus-indexed corpus. The prominence of Universidad de Castilla-La Mancha and the high productivity of Coral Calero and related collaborators suggest that Spanish dominance is likely driven by identifiable research groups that have consistently contributed to Green Software and software sustainability studies. Therefore, Spain should not be interpreted simply as a national-level phenomenon, but as evidence that a small number of specialized European research communities have shaped much of the visible literature.

The dominance of energy-related keywords, especially energy efficiency, energy utilization, and energy consumption, indicates that Green Software research is still strongly anchored in measurable technical performance. This is useful because energy can be observed, estimated, and optimized more directly than broader sustainability outcomes. However, it also shows that the maturity of the field remains uneven. The literature has developed a strong technical-energy orientation, but it is less mature in areas such as organizational adoption, governance, carbon accounting, developer

behavior, and integration with sustainable IT service management. In other words, the field has made progress in identifying what should be measured, but still needs stronger evidence on how Green Software practices can be institutionalized and sustained in real organizations.

Several themes remain underexplored. Carbon-aware software is still less visible than general energy-efficiency research, even though carbon intensity, workload shifting, and low-carbon deployment decisions are increasingly relevant for digital services. AI energy consumption also requires deeper attention because machine learning models, generative AI, and data-intensive applications can increase computing demand. Green DevOps remains an important future direction because sustainability metrics need to be integrated into continuous integration, continuous delivery, testing, deployment, and monitoring pipelines. Sustainable cloud services also require further study because cloud elasticity, virtualization, workload placement, and data center energy sources directly affect the environmental impact of software services. Organizational adoption is another critical gap because many studies focus on technical mechanisms but provide limited evidence on policies, incentives, capability building, and managerial decision-making.

For sustainable IT services, the findings can inform concrete practices. Organizations can translate Green Software principles into requirements engineering by defining energy, resource, and carbon-related quality requirements. During architecture and design, software teams can select patterns, algorithms, data structures, and deployment models that reduce unnecessary computation and infrastructure demand. During coding and testing, teams can include energy profiling, resource monitoring, and regression checks for energy-intensive functions. During deployment and operation, IT service managers can combine application monitoring with infrastructure indicators, cloud utilization data, and carbon-aware scheduling. In governance terms, Green Software metrics can be incorporated into service-level objectives, change management, capacity planning, procurement decisions, and sustainability reporting.

A stronger critique of the field is also necessary. First, Green Software measurement remains fragmented because different studies use different metrics, tools, workloads, and assumptions, making comparison difficult. Second, there is still a lack of widely accepted standards that connect software-level metrics with organizational

sustainability indicators. Third, empirical validation is often limited to prototypes, simulations, case-specific experiments, or small datasets, while large-scale validation in operational software organizations remains scarce. Fourth, adoption studies are still limited, so the field does not yet sufficiently explain how developers, managers, users, and governance structures influence Green Software implementation. These weaknesses show that Green Software Engineering still requires stronger methodological rigor, more comparable metrics, and deeper organizational evidence.

Compared with previous studies on Green Software, Green IT, and sustainable software engineering, the present findings confirm that energy efficiency and environmental impact remain central concerns. However, this study extends those discussions by combining SLR synthesis with bibliometric mapping, thereby showing not only what concepts appear in the literature but also how they are distributed across countries, institutions, sources, authors, and keyword networks. Previous conceptual and technical studies have emphasized energy-saving mechanisms, software design, measurement, and sustainability requirements. The present synthesis complements them by positioning those contributions within a broader map of research concentration, thematic clusters, and future service-oriented implications.

The use of the single keyword "Green Software" has important implications. On one hand, it provides a focused and traceable corpus that explicitly represents literature using the Green Software label. On the other hand, it may exclude relevant studies that use related terms such as sustainable software engineering, green coding, energy-aware software, carbon-aware software, sustainable cloud computing, or software sustainability. Therefore, the findings should not be generalized as a complete picture of all sustainability-oriented software research. They should be read as a Scopus-based map of publications explicitly indexed under the Green Software keyword strategy.

The country distribution also should not be overgeneralized as a complete global development pattern. Because the corpus is limited to Scopus-indexed, English-language journal articles and uses full counting of affiliation countries, countries with stronger publication visibility, international collaboration, and established research groups may appear more dominant. The imbalance of country contributions, especially the prominence of European institutions, indicates that Green Software research is not yet

geographically balanced. Future studies should therefore include broader databases, regional publications, and empirical contexts from developing countries to avoid reinforcing a narrow view of the field.

Table 6. Structured Synthesis of the Five Green Software Attributes

Attribute	Evidence from SLR and bibliometric pattern	Implication for sustainable IT services	Future validation needs
Energy efficiency	Strongly connected keywords include energy efficiency, energy utilization, and energy consumption; many studies examine software execution, algorithms, and design effects on energy use.	Include energy-related requirements, energy profiling, runtime monitoring, and optimization of energy-intensive software functions.	Validate with real workloads, production systems, and comparable energy measurement protocols.
Hardware efficiency	The literature shows that software choices influence hardware utilization, performance trade-offs, infrastructure demand, and resource efficiency.	Improve capacity planning, reduce over-provisioning, extend hardware life, and align software architecture with efficient infrastructure use.	Test across servers, cloud environments, mobile devices, and data center settings.
Measurement	Measurement appears as a recurring need because environmental claims cannot be verified without metrics, tools, and repeatable assessment procedures.	Integrate software energy, resource, and carbon indicators into service monitoring, SLA evaluation, and sustainability reporting.	Develop standardized metrics and compare tools across organizations and application types.

Attribute	Evidence from SLR and bibliometric pattern	Implication for sustainable IT services	Future validation needs
Climate commitments	The corpus links Green Software with sustainable development, Green IT, and climate-oriented digital transformation.	Connect software engineering decisions with organizational sustainability targets and climate-related governance commitments.	Assess whether Green Software practices measurably support institutional climate goals.
Carbon awareness	Carbon-related themes are emerging but less dominant than energy-efficiency themes, indicating a developing research frontier.	Use carbon-aware deployment, workload shifting, cloud-region selection, and carbon impact reporting for IT services.	Validate carbon-aware software practices with real carbon-intensity data and operational decision-making.

Future research should validate the five proposed attributes in real software organizations rather than relying only on conceptual synthesis or bibliometric evidence. Validation can be conducted through case studies, surveys, Delphi studies, controlled experiments, maturity assessments, or longitudinal monitoring in software companies, higher education institutions, government agencies, and cloud or data center environments. Such validation should examine whether the attributes are understandable for practitioners, measurable with available tools, relevant to service performance, and useful for guiding improvement actions. This step is necessary to move Green Software Engineering from a descriptive research area toward an operational evaluation and governance framework. Before such validation is completed, the five attributes should be treated as a synthesized analytical framework rather than as a fully validated maturity or operational evaluation model.

This study has several limitations. First, the dataset was taken only from Scopus, so relevant studies indexed in Web of Science, IEEE Xplore, ACM Digital Library, SpringerLink, ScienceDirect, Google Scholar, or regional databases may have been missed. Second, the search was restricted to the keyword "Green Software", which may exclude related literature using alternative terminology. Third, only English-language journal articles were included, which may introduce language and publication-type bias. Fourth, bibliometric counting may favor highly connected authors, countries, institutions, and keywords because the analysis used Scopus metadata and full counting. Fifth, keyword-based retrieval may include false-positive documents, as indicated by the need to treat the 2006 SuperState article cautiously. These limitations mean that the results should be interpreted as a focused Scopus-based representation rather than a complete global account of Green Software Engineering research.

Overall, the discussion shows that Green Software Engineering has clear relevance for sustainable IT services, but the field still needs stronger theoretical integration, standardized measurement, empirical validation, and organizational adoption studies. The bibliometric patterns reveal where the field has developed, while the SLR synthesis indicates how its concepts can be translated into practical service-oriented attributes. The next stage of research should therefore connect technical optimization with governance, carbon accountability, service management, and organizational capability so that Green Software can contribute more directly to sustainable digital transformation.

4. CONCLUSION

This study concludes that Green Software Engineering remains an emerging but increasingly growing research area and has clear relevance for sustainable IT services. Its main contribution is an integrated SLR-bibliometric map of Scopus-indexed Green Software Engineering research, which consolidates evidence on publication development, thematic structure, and practical implications without repeating detailed country or keyword results already presented in the Results and Discussion sections. Practically, the findings indicate that sustainable IT service planning can be strengthened by using Green Software attributes to guide energy-efficient application design, hardware-aware optimization, carbon-aware operation, measurement practices, and climate-aligned service improvement. The main methodological limitation is that the analysis relied only

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

on Scopus and the single search keyword "Green Software", so the findings should be interpreted as a focused Scopus-based representation rather than a complete account of all sustainability-oriented software research. Future research should expand the databases and search terms used in evidence retrieval and empirically validate the five proposed Green Software attributes in real software organizations and sustainable IT service environments.

ACKNOWLEDGMENT

Many thanks to the supervisors who provided support and direction for this research.

REFERENCES

[1] S. R. A. Ibrahim, J. Yahaya, H. Salehudin, and A. Deraman, "The Development of Green Software Process Model A Qualitative Design and Pilot Study," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 8, pp. 589–598, 2021, doi: 10.14569/IJACSA.2021.0120869.

[2] T. Kaur, G. Kumar, G. Singh, and V. Bhatnagar, "Green computing solutions an overview," in *Progressive Computational Intelligence, Information Technology and Networking*, 2025, pp. 86–90. doi: 10.1201/9781003650010-14.

[3] S. Li, H. Liu, B. Gong, and J. Wang, "An Algorithm Incarnating Deep Integration of Hardware-Software Energy Regulation Principles for Heterogeneous Green Scheduling," *IEEE Access*, vol. 8, pp. 111494–111503, 2020, doi: 10.1109/ACCESS.2020.3003304.

[4] M. Salam and S. U. Khan, "Risks mitigation practices for multi-sourcing vendors in green software development," *Proceedings of the Pakistan Academy of Sciences: Part A*, vol. 54, no. 1A, pp. 71–87, 2017.

[5] L. Singh, S. Tiwari, and S. Srivastava, "A study on creating energy efficient cloud-connected user applications using the RMVRVM paradigm," *Journal of Systems and Software*, vol. 213, 2024, doi: 10.1016/j.jss.2024.112033.

[6] H. Zhang and M. Ali Babar, "Systematic reviews in software engineering: An empirical investigation," presented at the Information and Software Technology, 2013, pp. 1341–1354. doi: 10.1016/j.infsof.2012.09.008.

[7] O. Poy, M. Á. Moraga, F. García, and C. Calero, "Impact on energy consumption of design patterns, code smells and refactoring techniques: A systematic mapping study," *Journal of Systems and Software*, vol. 222, 2025, doi: 10.1016/j.jss.2024.112303.

[8] D. Connolly Bree and M. Ó Cinnéide, "How Software Design Affects Energy Performance: A Systematic Literature Review," *Journal of Software: Evolution and Process*, vol. 37, no. 4, 2025, doi: 10.1002/smr.70014.

[9] A. Noureddine, M. D. Lodeiro, N. Bru, and R. Chbeir, "The Impact of Green Feedback on Users' Software Usage," *IEEE Transactions on Sustainable Computing*, vol. 8, no. 2, pp. 280–292, 2023, doi: 10.1109/TSUSC.2022.3222631.

[10] S. R. Ahmad Ibrahim, J. Yahaya, and H. Sallehudin, "Green Software Process Factors: A Qualitative Study," *Sustainability (Switzerland)*, vol. 14, no. 18, 2022, doi: 10.3390/su141811180.

[11] O. Danushi, S. Forti, and J. Soldani, "Carbon-Efficient Software Design and Development: A Systematic Literature Review," *ACM Computing Surveys*, vol. 57, no. 10, 2025, doi: 10.1145/3728638.

[12] E. Kern, M. Dick, T. Johann, and S. Naumann, "Green software and green IT: An end users perspective," *Environmental Science and Engineering (Subseries: Environmental Science)*, pp. 199–211, 2011, doi: 10.1007/978-3-642-19536-5_16.

[13] V. Majuntke, L. Hoffmann, F. Kronlage-Dammers, G. Beier, and N. Becker, "Green coding - bridging the gap from theory to practice," in *Uncertain Journeys into Digital Futures: Inter- and Transdisciplinary Research for Mitigating Wicked Societal and Environmental Problems*, vol. 1, 2025, pp. 165–172. doi: 10.5771/9783748947585-165.

[14] R. Vergallo, T. D'Alò, L. Mainetti, R. Paiano, and S. Matino, "Evaluating Sustainable Digitalization: A Carbon-Aware Framework for Enhancing Eco-Friendly Business Process Reengineering," *Sustainability (Switzerland)*, vol. 16, no. 17, 2024, doi: 10.3390/su16177789.

[15] D. Guamán, J. Pérez, and P. Valdiviezo-Diaz, "Estimating the energy consumption of model-view-controller applications," *Journal of Supercomputing*, vol. 79, no. 12, pp. 13766–13793, 2023, doi: 10.1007/s11227-023-05202-6.

[16] F. J. Rosa, J. C. de la Torre, J. M. Aragón-Jurado, A. Valderas-González, and B. Dorronsoro, "Energy–Performance Trade-Offs of LU Matrix Decomposition in Java Across Heterogeneous Hardware and Operating Systems," *Applied Sciences (Switzerland)*, vol. 16, no. 2, 2026, doi: 10.3390/app16021002.

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

[17] J. A. Larracoechea, S. Ilarri, and P. Roose, "A Proposal of Behavior-Based Consumption Profiles for Green Software Design," *Applied Sciences (Switzerland)*, vol. 14, no. 17, 2024, doi: 10.3390/app14177456.

[18] Y. Hu, T. Luo, N. C. Beaulieu, and W. Wang, "An initial load-based green software defined network," *Applied Sciences (Switzerland)*, vol. 7, no. 5, 2017, doi: 10.3390/app7050459.

[19] H. Noman, N. A. Mahoto, S. Bhatti, H. A. Abosaq, M. S. Al Reshan, and A. Shaikh, "An Exploratory Study of Software Sustainability at Early Stages of Software Development," *Sustainability (Switzerland)*, vol. 14, no. 14, 2022, doi: 10.3390/su14148596.

[20] D. Gil, J. L. Fernández-Alemán, J. Trujillo, G. García-Mateos, S. Luján-Mora, and A. Toval, "The effect of green software: A study of impact factors on the correctness of software," *Sustainability (Switzerland)*, vol. 10, no. 10, 2018, doi: 10.3390/su10103471.

[21] A. Fariña, O. Poy, N. R. Brisaboa, C. Calero, M. Á. Moraga, and Ó. Pedreira, "Onto how compression yields energy-efficient text search," *Computing*, vol. 107, no. 5, 2025, doi: 10.1007/s00607-025-01469-0.

[22] J. Liu and R. Lu, "Monitoring network through SNMP-based system," *International Journal of Intelligent Engineering and Systems*, vol. 5, no. 1, pp. 1–10, 2012, doi: 10.22266/ijies2012.0331.01.

[23] R. Vergallo, A. Cagnazzo, E. Mele, and S. Casciaro, "Measuring the Effectiveness of the 'Batch Operations' Energy Design Pattern to Mitigate the Carbon Footprint of Communication Peripherals on Mobile Devices," *Sensors*, vol. 24, no. 22, 2024, doi: 10.3390/s24227246.

[24] R. Vergallo and L. Mainetti, "Measuring the Effectiveness of Carbon-Aware AI Training Strategies in Cloud Instances: A Confirmation Study," *Future Internet*, vol. 16, no. 9, 2024, doi: 10.3390/fi16090334.

[25] A. González, J. Castaño, X. Franch, and S. Martínez-Fernández, "Impact of ML optimization tactics on greener pre-trained ML models," *Computing*, vol. 107, no. 4, 2025, doi: 10.1007/s00607-025-01437-8.

[26] M. Mohankumar and M. Anand Kumar, "Green database design model in software development life cycle phase," *Indian Journal of Science and Technology*, vol. 9, no. 30, 2016, doi: 10.17485/ijst/2016/v9i30/93118.

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

[27] P. S. Felix and M. Mohankumar, "Energy Optimization in Software Development: A Comparative Study of Sorting Techniques," *International Journal of Engineering Trends and Technology*, vol. 72, no. 7, pp. 339–349, 2024, doi: 10.14445/22315381/IJETT-V72I7P137.

[28] M. J. Page *et al.*, "The PRISMA 2020 statement: an updated guideline for reporting systematic reviews," *BMJ*, p. n71, Mar. 2021, doi: 10.1136/bmj.n71.

[29] N. Donthu, S. Kumar, D. Mukherjee, N. Pandey, and W. M. Lim, "How to conduct a bibliometric analysis: An overview and guidelines," *Journal of Business Research*, vol. 133, pp. 285–296, Sep. 2021, doi: 10.1016/j.jbusres.2021.04.070.

[30] N. J. Van Eck and L. Waltman, "Software survey: VOSviewer, a computer program for bibliometric mapping," *Scientometrics*, vol. 84, no. 2, pp. 523–538, Aug. 2010, doi: 10.1007/s11192-009-0146-3.

[31] F. Zhang, "SuperState: A computer program for the control of operant behavioral experimentation," *Journal of Neuroscience Methods*, vol. 155, no. 2, pp. 194–201, 2006, doi: 10.1016/j.jneumeth.2006.01.004.